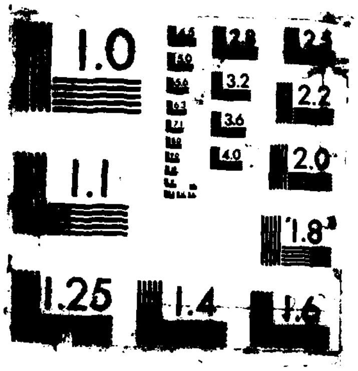
AD-A187 268 A GENERAL APPLICATION COMPUTER-ASSISTED INSTRUCTION SYSTEM FOR MICROCOMPUTERS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF SYST.. R MASON 1/3

UNCLASSIFIED SEP 87 AFIT/GLM/LSR/87S-45 F/G 12/5 NL


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

2

A GENERAL APPLICATION COMPUTER-ASSISTED INSTRUCTION SYSTEM FOR MICROCOMPUTERS

THESIS

Robert Mason
Lieutenant, Supply Corps, USN

AFIT/GLM/LSR/87S-45

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

2

AFIT/GLM/LSR/87S-45

A GENERAL APPLICATION COMPUTER-ASSISTED INSTRUCTION SYSTEM FOR MICROCOMPUTERS

THESIS

Robert Mason
Lieutenant, Supply Corps, USN

AFIT/GLM/LSR/87S-45

The contents of the document are technically accurate, and no sensitive items, detrimental ideas, or deleterious information is contained therein. Furthermore, the views expressed in the document are those of the author and do not necessarily reflect the views of the School of Systems and Logistics, the Air University, the United States Air Force, or the Department of Defense.

AFIT/GLM/LSR/87S-45

A GENERAL APPLICATION COMPUTER-ASSISTED INSTRUCTION SYSTEM FOR MICROCOMPUTERS

THESIS

Presented to the Faculty of the School of Systems and Logistics
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Logistics Management

Robert Mason, B.S.
Lieutenant, Supply Corps, USN

September 1987

Preface

The purpose of this study was to provide supply managers with an alternate method of augmenting aviation storekeeper training through the use of computer-assisted instruction on microcomputers. However, this effort evolved into a computer-assisted instruction system with a wide field of application. The system can be used in virtually any subject area to develop, administer, and monitor computer-assisted instruction techniques.

I would like to express my special appreciation to Major John Stibravy for his assistance, understanding, praise, and criticism during development of this training system. In addition to being my capable faculty advisor he was also friend, confidant, and colleague.

I would also like to express my appreciation to my wife and daughters for their understanding and support during the many hours required to complete this project.

Robert Mason

Table of Contents

List of Figures

Abstract

This study provides supply managers with an alternate method of augmenting Naval aviation storekeeper training to improve supply support management at operating sites. A microcomputer-based computer-assisted instruction system was developed which has much broader and significant application. The system can be used in any subject area to develop, administer, and monitor training and is applicable to all Navy, Air Force, Army, and other Department of Defense components. The potential for cost savings and improved operational capability through the use of this system is unlimited.

The system consists of two computer programs written in the BASIC programming language, program documentation, and a user's guide for the system. The system requires a minimum of computer operation knowledge but provides courseware of high quality and flexibility. The system was developed on a Radio Shack, TRS-80, Model 4 microcomputer and converted to operate on a Zenith, Z-248, IBM-AT/PC compatible microcomputer. These

The system creates and administers interactive computer-assisted instruction lessons. Each lesson can consist of any mixture of text screens, multiple choice question screens, and true/false question pages. Variable branching is allowed from question pages depending on student answer input. Lesson size (i.e., number of

screens in a lesson) is limited by program dimensioning to 200 screens and by available disk storage.

The courseware administration program, LEARNER/BAS, displays screens to the student based on courseware branching instructions. The program requires single key input by the student at the end of each screen (continue lesson or answer). On completion of each lesson use, the program records the student's name, date of lesson use, number of questions asked, number of correct/incorrect responses, and incorrect question numbers/incorrect responses to a disk file for analysis and lesson improvement by the courseware author.

The courseware authoring program, WRITE/BAS, is used to generate the text and branching table files for use by the courseware administration program. This program is fully menu driven with on-screen prompts. Using this program, a courseware author can create new lesson files, edit existing lesson files, print lesson files, and print student file reports.

The program documentation is complete so as to allow modifications and enhancements by the user (BASIC programming ability required) based on unique requirements or desires. The user's guide, although short, is complete and reflects the ease of program use and degree of user-friendliness.

A GENERAL APPLICATION COMPUTER-ASSISTED INSTRUCTION SYSTEM FOR MICROCOMPUTERS

I. Introduction

Background

In 1985, it was discovered on-board the aircraft carrier USS KITTY HAWK that $14 million worth of supplies could not be accounted for, sailors were able to requisition and receive 31 bars of pure silver, and some $5 million in F-14 fighter aircraft spare parts had been smuggled from the ship to Iran (17:18). This last incident prompted the Secretary of the Navy, John F. Lehman, Jr., to order an in-house (i.e., Department of the Navy) audit of supply management practices aboard aircraft carriers (7:4). The ensuing audit, conducted by the Naval Audit Service, revealed severe discrepancies in supply management practices aboard the carriers (7:4).

The typical inventory of an aircraft carrier's stock assets can amount to more than $300 million dollars and can be comprised of over 100,000 line items (4:4). During fiscal year 1985, almost $320 million in spare parts were improperly accounted for on aircraft carriers (4:4). "Officials since have declared that accountability will be as important as operational capability..." (4:4).

There are many effects of inaccurate inventory records and deficient supply management practices. As stated in a General Accounting Office report concerning an audit of inventory validity at Naval Supply Centers,

> Accurate inventory records are essential to the economic and effective supply support of U. S. military forces. Inaccurate records can result in critical supply shortages and prolonged delays in filling requisitions for materiel affecting mission readiness, inflated requests for funds, unnecessary expenditure of funds for procurement and repair of stocks, maldistribution of stocks, and accumulation and disposal of excess stocks. [10:Appendix, 1].

These deficiencies also indicate a loss of accountability for government-owned property and permit, perhaps encourage, outright fraud and use of government property for personal gain.

Among the findings of the Naval Audit Service investigation of aircraft carrier supply management practices, the most significant was the shortage of adequately trained aviation storekeepers aboard the ships (6:1). So significant, in fact, that "the auditors sharply criticized the Navy for permitting carriers to sail with an undermanned, inadequately trained force of aviation storekeepers" (4:4). In January of 1986, carrier manning of paygrade E-6 aviation storekeepers was 63% of that required, and manning of paygrade E-5 aviation storekeepers was 72% of that required (5:2). Actions were subsequently initiated to alleviate this manning shortfall and to upgrade aviation storekeeper training (5:2). "Officials stressed that efforts to improve carrier supply accounting procedures ... could mean fewer, better trained aviation storekeepers doing the job more effectively..." (5:2).

Aviation storekeepers are the Navy's enlisted occupational specialty for controlling aviation spares. They are responsible for the requisitioning, receipt, storage, inventory control, issue, and shipment of aviation material. They are also responsible for financial accounting of operating funds (flight hour funding and maintenance funding) and for interfacing with organizational and intermediate maintenance level personnel. They must effectively use a variety of manual and automated inventory and maintenance management systems to perform their functions.

Aviation storekeepers receive functional training in a variety of ways. Formal classroom education includes entry level training at Class A schools, advanced technical training at Class C schools, and specialty technical training at Class F schools. Non-classroom training is gained through rate training correspondence courses, on-the-job training, personal qualification standards completion, and self-study of publications contained in the Bibliography for Advancement Study. Ultimate responsibility "to train his subordinates in their own duties and in the duties to which they may succeed" belongs to the division officer (16:89).

Despite the training available, aviation storekeepers are not adequately trained to perform tasks required of them. The reasons for this include differing supply management systems, rapidly changing supply management systems and procedures, the nature of temporary additional duty manning of supply activities, and difficulties in comprehension of supply manuals and directives. Additionally, inadequate manning levels lead to shortfalls in training.

The Navy uses three different supply management mechanized systems and at least three different aviation supply/maintenance mechanized systems (15:Ch 4, 22). Each of these systems is generally in a state of flux with procedural changes caused by program updates or local embellishments. In essence, a senior experienced aviation storekeeper transferred to a new duty station in a supervisory capacity may have little knowledge of the supply management system in use and may require considerable time to develop sufficient knowledge to function effectively.

The problem is compounded by the concept of temporary additional duty manning in aviation support divisions at naval air stations and aboard aircraft carriers. Under this concept, aviation storekeepers are assigned to an aviation squadron and are temporarily assigned to either the naval air station or ship from which the squadron is operating. The intent of this policy is to automatically compensate for the changes in supply workloads brought about by deployment of an aviation squadron. An aircraft carrier with an embarked air wing will typically have about one-half of the total aviation support division manning comprised of squadron aviation storekeepers who had previouslly been assigned to perhaps four different naval air station aviation support divisions. Many aviation storekeepers are thus faced with different operating systems, new procedures, and in all probability a new job assignment virtually overnight. The manager is faced with the influx of many new personnel of unknown capabilities and experience. He must incorporate them into his organization and be capable of providing support during high tempo operations instantaneously. The aviation support division's ability to provide a high degree of supply support is critical to the readiness of the embarked aircraft and the warfighting capability of the aircraft carrier. The magnitude of the training problem for these personnel is obvious.

Another factor contributing to inadequate aviation storekeeper training is the readability level of supply manuals and related publications. Although rate training correspondence courses and curricular reading materials are designed with readability in mind, supply publications are not. To an aviation storekeeper of average intelligence, the task of reading and retaining any portion of a lengthy manual written well above his readability level must seem insurmountable. The problem is multiplied if he lacks adequate reading skills or if English is a second language as is often the case in the aviation storekeeper rating.

Yet another factor contributing to inadequate aviation storekeeper training is the aviation storekeeper manning shortfall itself. Despite its false economy, training is often the first area neglected when manning is inadequate. In an effort to get replacement personnel to a duty station as rapidly as possible, enroute formal schooling may be foregone. When longer working hours are required to perform daily tasks, it is difficult to devote additional hours to formal training or self-study.

The final factor in the aviation storekeeper training "formula for failure" is the self-perpetuating nature of the problem. Senior aviation storekeepers lack the requisite knowledge to properly train subordinates. Performance deficiencies cause errors which must be corrected which require additional expenditure of limited man-hours which leads to further declines in training.

Statement of the Problem

Aviation storekeepers are not adequately trained to perform the tasks they are expected to perform, and currently utilized training methods are not sufficient to provide the required expertise. This affects the quality of supply support afforded an embarked aviation wing and its subsequent operational readiness, which ultimately impacts the carrier's warfighting capability.

Purpose of the Study

The purpose of this study was to provide supply managers with an alternate method of augmenting aviation storekeeper training so as to improve the support and supply management at operating sites. It involved the development of a computer-assisted instruction technique to be applied at the operating site using existing microcomputers and user written training modules. To that end, the following research objectives were established:

Research Objective Number One. Develop a microcomputer program to generate computer-assisted instruction modules for the training of aviation storekeepers. The program should require a minimum of computer programming expertise on the part of the module developer, no programming expertise on the part of the user (i.e., trainee), and be adaptable to a variety of microcomputer types.

Research Objective Number Two. Develop documentation for the computer-assisted instruction programs to permit modification by users if desired.

Research Objective Number Three. Develop user's guides to provide instructions for using the computer-assisted instruction programs.

Scope and Limitations of the Study

With the time and resource limitations for completing this research project, extensive field testing of the programs was not feasible. The programs do permit the development and administration of computer-assisted instruction courseware. Additional testing could best be accomplished by providing the program and documentation to operating sites for use as desired by management personnel.

The programs developed could not be tested on all possible microcomputers. It was developed and its operation demonstrated on a Radio Shack, TRS-80, Model 4 microcomputer since that machine was readily available to the researcher. To evaluate the programs' transportability, it was subsequently converted and demonstrated on a Zenith, Z-248 microcomputer (IBM AT/PC compatible) since this is rapidly becoming the Department of Defense standard microcomputer. Conversion of the programs to other microcomputers is straightforward with BASIC programming knowledge.

Applicability

Although the motivation for undertaking this research project was to provide supply managers with an alternate method of augmenting aviation storekeeper training, the computer-assisted instruction system which evolved has much broader and significant application. The system can be used in any subject area to develop, administer, and monitor computer-assisted instruction techniques and is applicable to all Navy, Air Force, Army, and other Department of Defense components.

This computer-assisted instruction system was demonstrated to the Air Force Institute of Technology, School of Systems and Logistics, Department of Exportable Education and is currently under review for use. The potential cost savings from this application alone are substantial. These cost savings would be realized from the following three major advantages of this computer-assisted instruction system:

1. Since many exportable education subjects could be developed for use with this system, travel and lodging at operation sites would be reduced by thousands of dollars annually.

2. Managers at operational sites could administer course material as time and circumstances allow rather than disrupting day-to-day operations for personnel to attend formal classes. With increasing ownership of home computers, effective voluntary training could even occur during off-duty hours.

3. This form of training would be available to operational sites when the need exists -- not only when a site training team can arrange a visit. The addition of this flexibility to meeting training requirements would increase any operating site's effectiveness and contribute to increased combat capability.

Another advantage of this computer-assisted instruction system is the dynamic nature of the programs. The system permits modification to meet individual needs, encourages improvements to the system, and provides feedback mechanisms to courseware authors to allow improvement of courseware material. These modifications and improvements would be shared with other users, thereby increasing the overall effectiveness of the training. Additionally, the potential is unlimited for operating sites to share the courseware developed using the system.

The development of this computer-assisted instruction system affords DOD managers an unheralded degree of flexibility in meeting training requirements while decreasing training costs and increasing training effectiveness. The system enables experts to convey knowledge in a straightforward, simple manner without detriment to daily operations.

II. Literature Review

> The computer is incredibly fast, accurate, and stupid. Man is unbelievably slow, inaccurate, and brilliant. The marriage of the two is a force beyond calculation [12:42].

Today's military and civilian managers seem to have taken the above quote (attributed to Leo Cherne, famed economist, lawyer, and sculptor) very much to heart. Computers are used in virtually every facet of administration, planning, and production. Applications include electronic word processing, mechanized supply and inventory records, computer-aided design, computer-aided drafting, mechanized billing, and numerically controlled machines. One application gaining more and more attention in the management world is that of computer-assisted instruction.

Kearsley and Hillelsohn, noted experts in computer-based education, conducted a survey of 200 randomly selected industry training managers to determine the status of computer-based training use. They found that forty-two percent of those responding were using computers in regular training activities and forty-one percent were exploring the use of computers. Computer-based training can be used to teach management/supervision skills, technical tasks, field engineering, policy rating, data processing/programming, new employee orientation, computer-managed instruction, business simulations, process control, equipment maintenance, financial skills, machine operation, computer literacy, basic skills, and to conduct testing of student progress. From this list, they concluded that "just about any training application is suitable for [computer-based training]" (13:21+).

Dr. Alfred Bork, Director of the Educational Technology Center at the University Of California, Irvine, predicts that "computers will comprise the dominant delivery system in education for almost all age levels in most subject areas" over the next 25 years (2:4). The technological implications of computer learning on education are comparable to those of the invention of the printing press (2:1). However, Dr. Bork is quick to emphasize that each application of computers in educational contexts should be justified since education is not automatically improved using current computer learning methods (2:5).

Based on these observations, it was decided that computer-assisted instruction techniques could be successfully used to augment aviation storekeeper training. However, some background knowledge of the use of computers in education and training is required to effectively apply computer-assisted instruction techniques and to avoid common pitfalls in its use.

The remainder of this chapter provides an introduction to the use of computers in education and training with particular emphasis on computer-assisted instruction techniques. Terms associated with computer-based education are defined and modes of computer-assisted instruction are described. The advantages and disadvantages of using computer-assisted instruction in training programs are discussed. Considerations of the components of a computer-assisted instruction system (i.e., hardware, software, and courseware) are then presented.

Computers in Education and Training

The computer industry was probably the first group to use computer-based training beginning in the late 1950s (9:20). Although this application was used to teach about computers, the following discussion is applicable to subjects other than the computer itself. Before addressing computer-assisted instruction in particular, a discussion of the uses of computers in education and training and definitions of common terms is warranted.

Figure 2.1 illustrates some of the more common terms used in computer-based education and training. Computer-based education is an "umbrella" term which encompasses all uses of computers in conjunction with education. It is comprised of three major elements - computer-managed instruction, computer-assisted instruction, and computer-supported learning resources. Terms used synonomously for computer-based education include computer-based learning and computer-based training. Computer-based learning is the term most often applied to academic applications while computer-based training is the term most widely accepted in industrial applications.

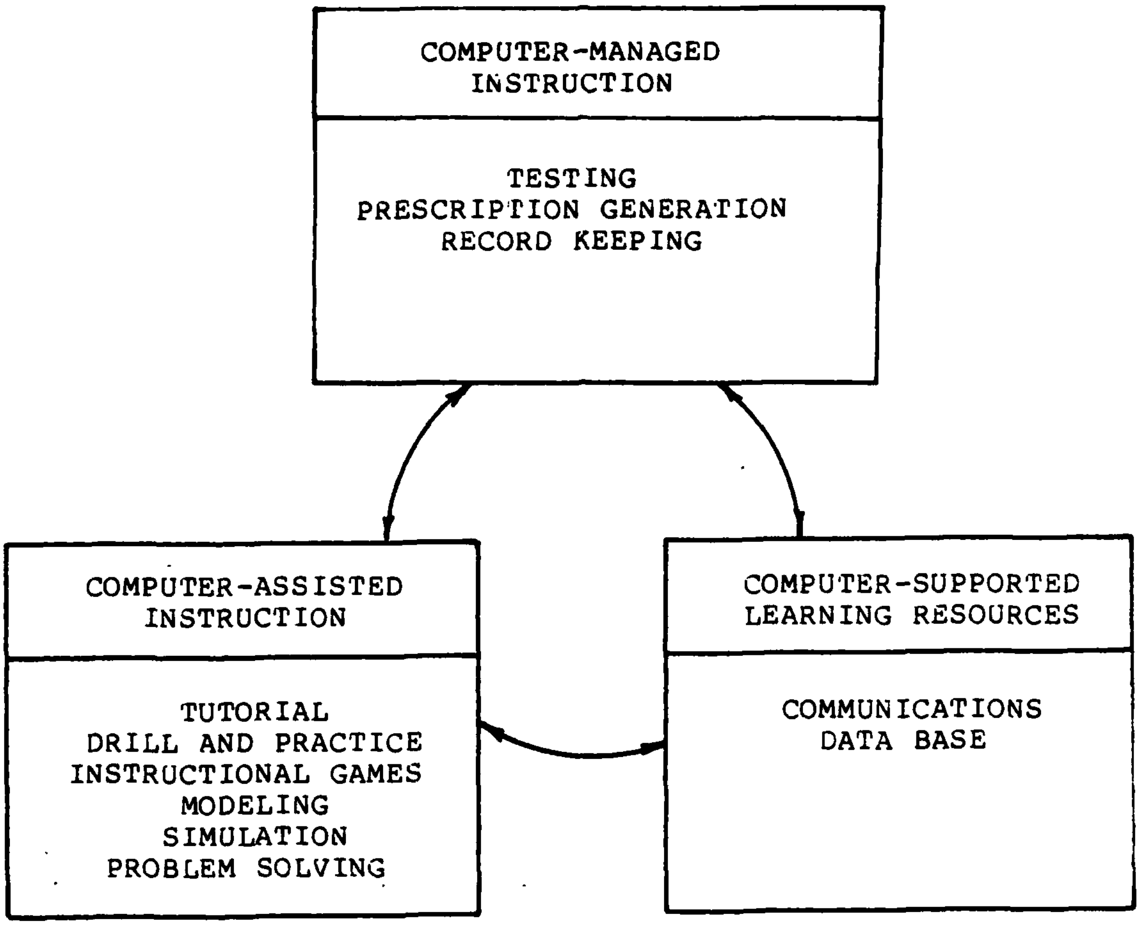


Figure 2.1: Computer-Based Education Applications

Computer-managed instruction is the management of training or education using computer resources. Its use may be invisible to the student as no actual computer interaction is necessary to constitute computer-managed instruction. This use of computers in education consists of three interdependent elements - testing, prescription generation, and record keeping. Testing measures a student's knowledge in a particular subject area (pre-training knowledge) or evaluates a student's comprehension of training objectives (post-training knowledge). Angus Reynolds, a senior human resources development consultant with Control Data Corporation, considers testing the foundation of any computer-managed instructional system since it "provides the information needed to prescribe 'learning activities'" (19:35).

The prescription generation element of computer-managed instruction encompasses the analysis of test results and the determination of what training is required to improve the student's mastery of the subject area. This most frequently means generating a list of training activities (courses, programmed instruction modules, and computer-assisted instruction modules) which the student must complete. This element is the key to computer-managed instruction's power. Each student is required to complete only that training required to attain mastery of the knowledge area.

The record keeping element of computer-managed instruction "automatically generates and stores records of individual and group progress" (19:35). This record keeping function permits recall, manipulation, and display of data needed to analyze training progress for any selected group. The primary advantages of this element of computer- managed instruction are the elimination of manual record keeping and elimination of printed data which may not be required for analysis.

Computer-assisted instruction is the use of the computer in an actual instructional process (i.e, as an instructional medium). In its simplest form, computer-assisted instruction consists of little more than mechanized programmed instruction. Visual aids afforded by computer graphics range from non-existent to very complex depending on the subject material being taught, the availability of graphics on the computer being used, and the software package used. Current trends in computer-assisted instruction include adoption of multimedia techniques in which the computer is used in conjunction with taped video material, video discs, photographic slides, or other visual material. Computer-assisted instruction techniques will be covered in greater detail in the following section.

Computer-supported learning resources is the term applied to a system for information storage and retrieval (data base) or instructional communications. As such, it neither teaches nor performs management functions. It does, however, permit the exchange of information among users, sharing of common data among users, and mechanized interaction between instructors and students (19:37).

It should be noted that computer-managed instruction, computer-assisted instruction, and computer-supported learning resources are independent entities which may or may not be used in conjunction with one another. For instance, a computer-managed instructional system can be implemented without students actually using a computer, or computer-assisted instructional techniques can be used without using computer-managed instruction or computer-support learning resources systems. However, the three systems can be easily used together in one large education and training package. As an example, a student may take a test at a computer terminal and have training modules prescribed by the program (computer-managed instruction), complete training modules on the computer (computer-assisted instruction), and receive feed-back from an instructor on the computer (computer-supported learning resource).

Of these three major uses of computers in training, the one most applicable to augmenting aviation storekeeper training is computer-assisted instruction. However, computer-assisted instruction can be applied in a variety of methods as was illustrated in Figure 2.1. Therefore, an in-depth discussion of these methods is warranted.

Computer-Assisted Instruction Techniques

"Computer-assisted instruction grew out of the technology of programmed learning (itself a derivative of the Socratic method, seasoned with Skinner)" (8:22). In one simple form, computer-assisted instruction is little more than a mechanized, programmed instruction in which the student is presented with a portion of text, asked a question about the text he has just read, and a new portion of text is presented based on his answer. Computer-assisted instruction is, however, this and much more. Reynolds lists six distinct modes of computer-assisted instructional techniques - tutorial, drill and practice, instructional games, modeling, simulation, and problem solving (19:35). Each of these methods warrants further discussion.

The tutorial method is that technique described above as an off-shoot of programmed instruction. The primary purpose of this method of computer-assisted instruction is to impart new knowledge to the student. The computer's advantage in presenting this type of material is that it is interactive, requiring participation by the student throughout the training exercise. The student cannot skip portions of text and the program will not advance to the next block of data until the student has demonstrated at least minimal understanding of the material presented.

Drill and practice techniques are most analagous to flash cards or other repetitive exercises. The student is asked a series of questions, the answers are evaluated, and the student informed of progress throughout the lesson. This technique is most applicable to teaching simple, repetitive skills (mathematics, spelling, and typing) but can be extended to very complex procedures such as maintenance troubleshooting procedures, writing skills, business transaction analysis, and management or supervisory skills. In some computer-assisted instruction programs (especially those dealing with developing basic mathematical skills), the computer attempts to evaluate the possible source of erroroneous responses and provides corrective, remedial instruction.

Instructional games seek to impart knowledge under the auspices of entertainment (19:36-7). The most familiar games teach rudimentary skills (arithmetic, word association, and spelling). More sophisticated, higher level concepts can be taught such as wargame programs which teach military tactics. Another application of instructional games in computer-assisted instruction is their use in conjunction with other techniques to provide a motivational reward at the end of the training session for performance during the lesson. Oftentimes, the student is able to achieve superior performance in the game portion of the lesson using the knowledge gained during the learning portion of the lesson.

Modeling and simulation techniques are closely related and offer perhaps the greatest potential for industrial and military applications. They attempt to represent a system or process in which the student can change values and observe the results of his actions (19:37). Modeling implies no attempt to create realism; simulation attempts realism through graphic display or hybrid forms of equipment (19:37). Hillelsohn, a noted courseware development manager, describes the advantages of modeling and simulation as follows:

> ...simulation offers the opportunity to actually practice a skill. The student can make wrong decisions and get appropriate feedback about the consequences without the dangers and costs associated with real equipment. Computer-based simulation also allows the process being studied to be stopped, started, reviewed, slowed down, speeded up, or altered to enhance understanding [12:42].

It is apparent that modeling and simulation are applicable to numerous industrial and military situations, especially in developing skills in manufacturing processes or casualty control procedures for which actual practice is expensive, dangerous, or rarely occurs naturally (12:42).

The final computer-assisted technique, problem solving, entails the use of the computer by the student to solve problems as a means of knowledge achievement (19:37).

Each of these modes of computer-assisted instruction could be used advantageously to augment aviation storekeeper training. The mode selected depends on the primary intent of the instructicn. For instance, tutorial methods could be used to impart new knowledge of supply operations, drill and practice methods could be used to practice repetitive operations such as material identification, modeling and simulation could be used to simulate actual supply operations. Since the intent of this research project was to develop a method of imparting new knowledge to aviation storekeepers, the tutorial mode of computer-assisted instruction was selected as the most effective means of achieving this goal.

Advantages and Disadvantages of Computer-Assisted Instruction

Before discussing the components of a computer-assisted instructional system, it is helpful to review the advantages and disadvantages of the use of computer-assisted instruction. In separate articles, Govaerts and Grillot (11:28-9), Walker (22:39+), and Bork (2), noted experts in the field of computer-based learning, individually discuss advantages and disadvantages of computer-assisted instruction. However, Dr. Margaret Bahniuk's list (1:85) is comprehensive and encompasses their thoughts. Dr. Bahniuk, Associate Professor of Business Education at Cleveland State University, includes the following advantages of computer-assisted instruction:

1. "The process is interactive; the student has an active role in the learning process.
2. The process is flexible and consistent. Each student learns at his own pace and studies only that material required to gain subject mastery.
3. The student gets instant feedback on performace during the training process.
4. A reduction in actual equipment needs by graphics capabilities allowing simulations.

5. A reduction in total time needed for training (as a result of prescribed learning evolutions).

6. A corresponding reduction in total training costs."

She also lists the following disadvantages:

1. "Time may be insufficient for implementing or revising software.

2. Limited computer equipment may lead to scheduling problems.

3. Software may be unreliable.

4. System response time (elapsed time between student computer input and computer response) may be inadequate.

5. Some people may learn best in other environments (classroom, self-study, or personal interaction)."

Of the above advantages, the reduction in training time and its associated cost savings is notable. "An average time saving of one-third is typically found in comparing computer-based education programs with conventional ones" (22:40).

Despite the seemingly overwhelming arguments for using computer-assisted instruction, Reynolds (19:38) and Walker (22:41) are careful to note that computer-assisted instruction cannot totally replace conventional training, but rather can supplement these techniques. Computer-assisted instruction should be used in those areas where it is the better training technique, but not attempted in those cases where it is clearly not as capable.

Another major drawback to computer-assisted instruction, as noted above, is the time and cost constraints of developing and maintaining/updating training modules (22:42). Highly complex, multi-media training modules are very expensive (as much as $50,000 to $500,000) (14:136), but perhaps worth the expense, if the training is not available using conventional techniques or if an overall cost savings is realized from not training on actual equipment. However, the expense of developing tutorial modules should be minimal and more than offset the savings in training time for computer-assisted instruction to be beneficial.

Assuming that the manager decides to implement some form of computer-assisted instruction to augment his training program, the next step is selection of a system to present the material. The following section will describe the components of a computer-assisted instruction system and present some of the considerations for each of the components.

Computer-Assisted Instruction Systems

Figure 2.2 illustrates the three basic components of a computer-assisted instruction system - hardware, software, and courseware. Note that the final two components can be integrated or separate. Each of these components will be discussed in greater detail.

The hardware component is the computer itself and any peripherals required for data input, output, or storage. The computer may be a mainframe computer with interconnected terminals (centralized or timeshared system) or a

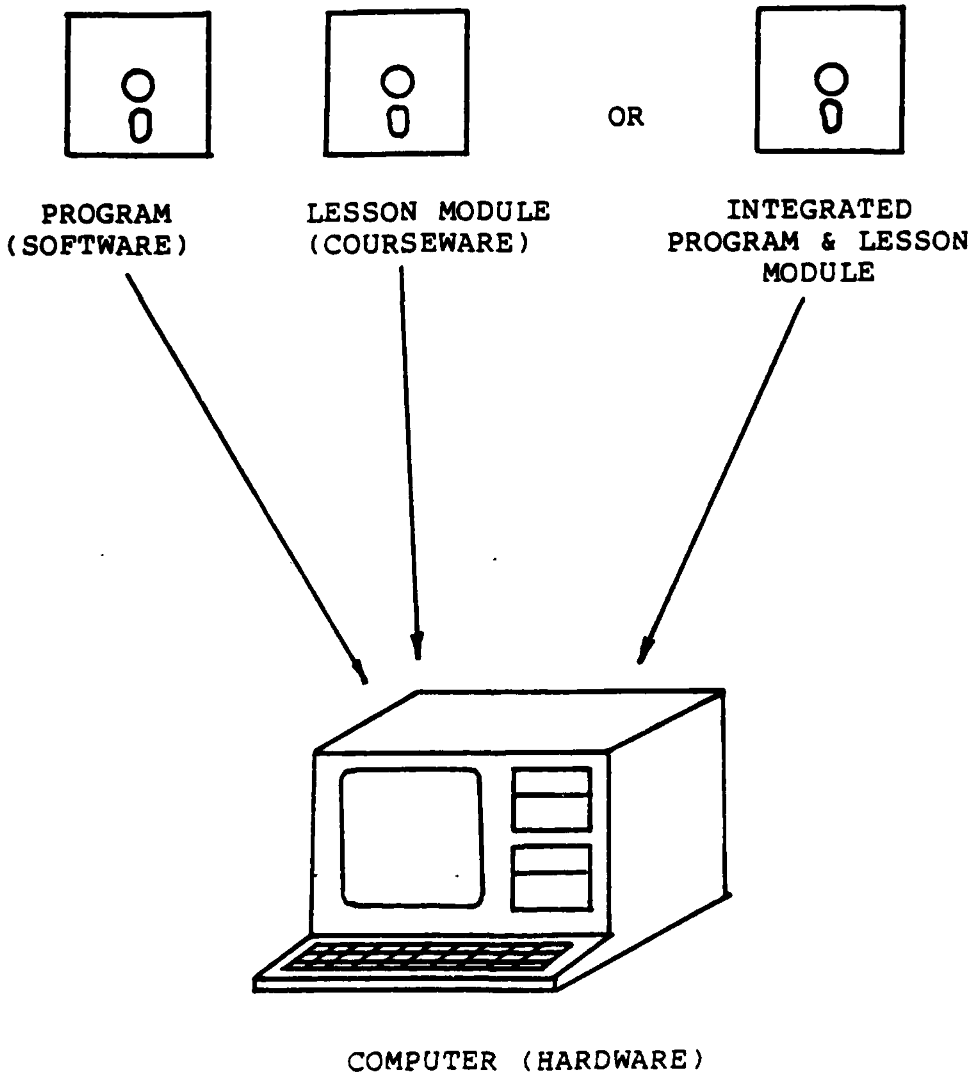


Figure 2.2: Computer-Assisted Instruction System Components

standalone microcomputer. The hardware chosen for a computer-assisted instruction system depends on the application or the extent of training to be conducted on the system. However, Reynolds and Davis argue against using existing mainframe computers for training purposes for two key reasons - controls over access to operations computers and limited computer power of these computers (20:45). Therefore, the standalone microcomputer was chosen as the hardware selection for this research project.

Software is the program which enables the computer to operate. For computer-assisted instruction purposes, software is the program which operates the computer to present the instructional material, branch to different portions of the material, and record information from the training period. Common programming languages for computer-assissted instruction systems include Pascal, BASIC, CP/M, C, and FORTH.

Courseware is the instructional material itself including text, questions, answers, and branching instructions. It can consist of formatted ASCII text files or can be incorporated in the software as is commonly found in simple BASIC language computer-assisted instruction systems. Courseware can be developed using common text editors (i.e., word processing programs) or written with the help of a courseware authoring program, as was done in this research project.

Of these three computer-assisted instruction system components, the most important from an effectiveness perspective is the courseware. Boyd and Eldridge (3:36+), Fauley (8:22+), and Hillelsohn (12:42+), noted experts in the field of computer-based education, each discuss the importance of quality courseware when implementing computer-assisted instruction techniques. Poorly written, boring, or incorrect instructional materials are more detrimental when used in conjunction with computers than in other mediums. Boyd and Eldridge in particular argue that courseware ergonomics (the relationship between learners and the computer; human factor considerations) whould be of primary importance in computer-assisted instruction methods (3:38). They state, "It is unfortunate that many print aids have not been adapted in [computer-assisted instruction] ... learners need even more help when using an unfamiliar medium" (3:38). The print aids to which they refer include simple sentence structure, active voice, shorter sentences, shorter paragraphs, and adequate use of blank space (3:38).

To further enhance the effectiveness of computer-assisted instruction, Boyd and Eldridge maintain that the programs must be as user friendly as possible and that the trainee must understand how to use the program. The quality of the courseware is of little consequence if the trainee does not understand how to turn the computer on, load the training module, operate the keyboard, exit the program, and turn the computer off (3:38-9). For those people already apprehensive of computers, lack of this requisite knowledge alone can doom the training program to fail.

For the purposes of this research project, the computer-assisted instruction system was somewhat predetermined. For reasons cited elsewhere, the hardware consisted of a standalone microcomputer system. The software consisted of a product of this research and was written in the BASIC computer language (note, though, that the program could have been compiled and run in assembly or machine language). The courseware was developed using another product of this research, a BASIC language courseware authoring program, to be used in conjunction with the software program.

III. Methodology

General Method

Two microcomputer programs were developed using the BASIC programming language. The first program, WRITE/BAS, functioned as a courseware development aid; the second, LEARNER/BAS, functioned as a courseware administration program. Documentation for each of the programs was then developed. Finally, instructions for using each of the programs were developed in the form of a user's guide.

Specific Procedures

The elements of this computer-assisted instruction system were developed on a Radio Shack, TRS-80, Model 4 microcomputer. This microcomputer is a Z-80 based high speed microprocessor with 64K of memory (18:A-47) and two 5.25 inch single-sided floppy disk drives. The microcomputer operates under the TRSDOS (Tandy Radio Shack Disk Operating System) Version 6 operating system (copyright 1983, Logical Systems).

Step 1. Since a specific microcomputer was available, the first step in developing this microcomputer program was the selection of a programming development language. Bruce Tonkin, a noted software developer and computer industry critic, compares BASIC, Pascal, and C using a 100 point scale in each of ten equally weighted categories (easy to learn, handles typical data types, ease of disk-file read/write, access to other languages, access to hardware or disk operating system, capability of input/output to standard devices, includes or allow graphics, contains transcendental functions, has standard syntax, and capable of modularity support). His assessment of BASIC as being the superior of these three languages and its particular applicability to this programming project led to the selection of BASIC as the development language for this program (21:96+). The specific language used was BASIC for TRSDOS Version 6 (copyright 1983, Microsoft).

Step 2. The next step was the development of block diagrams for each of the programs. The block diagrams depict the program logic and serve as program development tools.

Step 3. The next step was the writing of the programs in BASIC code. A modular approach was used to facilitate program efficiency and modification. Program operation was fully documented using remarks within the code.

Step 4. The next step involved program testing and debugging. Since a modular programming approach was used, this task was eased considerably - each module could be tested and debugged individually. Several training modules were developed using the program, and test runs of the training program were conducted to demonstrate their proper functioning. One of the training modules developed serves as an introductory lesson which can be executed from the LEARNE BAS program.

Step 5. The next step entailed modifying the programs to operate on a different type of computer to demonstrate the programs' transportability. The program was manually modified to operate on a Z-248, IBM AT/PC compatible microcomputer under GW-BASIC. Each of the preceding steps were in support of the first research objective.

Step 6. The final step consisted of developing the written documentation and guidelines for using the programs to administer computer-assisted instruction. This step was in support of the second and third research objectives.

IV. System Development

The products of this research project are presented in Appendices A through F of this report. Appendix A contains the documentation and program listings for LEARNER/BAS, Appendix B illustrates operation of the program, and Appendix C contains the user's guide for this program. Appendix D contains the documentation and program listings for WRITE/BAS, Appendix E illustrates operation of the program, and Appendix F contains the user's guide for this program.

In addition to this report, computer disks containing the programs (individual disks for Tandy Model IV and IBM PC/AT compatible versions) and a separate user's guide for the system were produced. At the time this research report was submitted, a distribution scheme for disks and copies of the user's guide had not been developed. Parties interested in obtaining these products are encouraged to contact the author at the following address: LT Robert Mason, c/o C. L. Mason, 721 Sunburst Lane, Dallas, TX 75218.

The remainder of this chapter discusses development of these programs and the user's guide.

Program Development

As discussed in the preceding chapter, the development of this computer-assisted instruction system basically followed standard programming procedures (developing block/flow diagrams, encoding the programs, and testing/

debugging the programs). Although a total "top-down" approach is enviable, the author found the programs to be very dynamic from the beginning of development. As modules were written and tested, enhancements were identified and added to the program. The dynamic nature of this system will be discussed in greater detail in a later section of this report.

In essence, WRITE/BAS and LEARNER/BAS comprise a specialized data base management program with word processing features. Blocks of records form a lesson screen. During lesson execution, the student's computer input at the end of the screen determines the next block of records to be displayed. WRITE/BAS is a courseware development tool with limited word processing features for creating the data base which LEARNER/BAS will use to present the lesson to a student.

The program was envisioned as displaying formatted screens to the student/courseware author throughout the lesson. The top line of the screen provided information about the lesson while the bottom line of the screen provided instructions/prompts for using the program. Twenty lines between these lines contained the text material to be presented. Examples of this screen "boilerplate" are illustrated throughout Appendices B and E. The author felt that three general types of screens (i.e., text, multiple choice question, and true/false question) would allow lesson development and training potential.

The next step in development of this system was design of the data base holding the lesson text records. It was clear that lesson lengths would exceed the memory capacity of the Model IV and would therefore require storage on disk. Data records were accessed as necessary during lesson execution. Consideration then turned to whether records should be stored as a sequential ASCII file or as a formatted, direct-access file. Consideration of disk storage efficiency (amount of information that can be stored on a single disk) would favor a sequential file. However, direct-access files allow much quicker access to specific records within the file. Since a major advantage of computer-assisted instruction is selective branching within a lesson and since prolonged computer response times (i.e., elapsed time from student input to computer response) can be distracting to the learning process, a direct-access file was chosen as the format for storage of lesson material. The record file formats for this file are contained in Appendices A and D.

Early in the design process it was realized that searching through a lengthy direct-access file for a particular lesson page number would be highly inefficient and severely degrade computer response time. The text file is, therefore, read once and information from the header records extracted (record number, page number, and number of additional records forming the page) to form a lesson table on the disk in a sequential ASCII format. This table is then read into the computer memory during lesson initialization. This procedure allows very rapid identification of the file records to read and display during lesson execution. Again, the elements of this file are contained in Appendices A and D.

The final disk file used by this system is a student file containing results of lesson execution. The student's name, date lesson executed, number of responses requested, number of correct and incorrect responses, and question numbers/incorrect responses are stored in memory and written to a disk file on lesson termination. This data is printed using a portion of the WRITE/BAS program. The elements of this file are contained in Appendices A and D and a sample of the resulting report is illustrated in Figure E.6. Although the student file report calculates and prints a percentage grade for the lesson, this grade is printed for use by courseware monitors primarily for lesson modules comprised totally of questions (i.e., tests). For instructional lesson modules, the grade and incorrect questions/responses reflect the quality of the courseware. For instance, a question missed repeatedly indicates a shortcoming in instruction or question construction.

Once data base requirements were established, the two programs were developed to create and use this data base structure. Program organization (a personal preference of the author) was such that subroutines are physically located at the beginning of the program with the main program following. Both programs were documented throughout development with liberal remark lines.

LEARNER/BAS was first program of the two to be developed. It was developed on the Tandy Model IV microcomputer. Conversion of the program for operation on an IBM PC/AT compatible computer will be discussed later. Structure of this program and additional comments regarding program operation are documented in Appendix A of this report. WRITE/BAS was then developed and proved to be the more complicated program. Structure of this program and additional comments regarding program operation are documented in Appendix D of this report.

Since eventual conversion of the programs to IBM PC/AT compatible microcomputer operation was desired, several precautions were taken in program development. First, no PEEKS or POKES (i.e., manipulation of memory outside BASIC commands) were allowed. Second, no machine/assembly language subroutines were allowed. Third, disk drive designations were assigned in single line variable assignments rather than in each OPEN statement which would require fewer changes during conversion. These constraints did not prove to be a hinderance in program development.

Both programs were tested and debugged initially by the author. Improvements and enhancements were noted during testing and incorporated into the program code. The author attempted to "crash" the programs using both logical and illogical means and error routines/corrections developed where necessary. During this phase of program development, a tendency to attempt protection against any eventuality was noted. It was necessary to balance the possibility of erroneous input against the additional coding required to provide protection. All reasonable errors have been protected in the current versions of the programs.

The programs were then extensively tested by an adult high school graduate and one teenager. Since neither had significant computer experience, these test cases represented typical skill levels of military and civilian DOD personnel. Each successfully used the programs with no direction from the author. Problem areas were observed and changes incorporated or noted for inclusion in the user's guide. The programs were then demonstrated to several AFIT graduate students for additional evaluation.

Program conversion was accomplished by uploading the BASIC programs in ASCII format to the Air Force Institute of Technology mainframe computer and then downloading them to the Zenith Z-248 microcomputer. This effectively converted the programs from a TRSDOS formatted disk to an MS-DOS formatted disk.

Conversion from this point was straightforward due to the similarities between the two versions of Microsoft BASIC used. However, four differences existed between the two versions which affected conversion of these programs. First, drive designations for the data files had to be changed (TRS-DOS version designates drives numerically after the file name while MS-DOS designates drives alphabetically before the file name). Second, the method of locating the cursor on the screen required change. The Tandy version uses "PRINT@ (xx,yy)" (where xx is line position and yy is column position) to locate the cursor while the MS-DOS version required "LOCATE xx,yy" to position the cursor. Third, one non-standard ASCII character (ASCII value 31) required emulation in the MS-DOS version. In the TRSDOS version, ASCII value 31 erases the display to the end of the current line. This was emulated on the Zenith microcomputer by the commands "LOCATE xx,yy:PRINT STRING$(81-POS(0),32): LOCATE XX,YY". These commands position the cursor, print blank characters to the end of the line, and then reposition the cursor to the original location. Fourth, the TRSDOS version begins numbering lines and columns with zero, while the MS-DOS version begins with one.

Following conversion, the program was tested on the Zenith Z-248 microcomputer using the same procedures as described earlier. No changes were required with the exception of the above mentioned differences in BASIC versions.

Documentation Development

Following program development, debugging, and testing, documentation for the programs was generated. This program documentation is included in Appendices A and D of this report. Included are program overviews; block diagrams of program operations; variable lists; cross-reference listings by line number, variable name, and BASIC keywords; and the program listings. This complete program documentation is provided for those who may wish to modify or enhance the programs.

User's Guide Development

The final step in the development of this computer-assisted instruction system was the writing of the user's guide to provide instructions regarding use of the system. The results of this effort are contained in Appendices C and F of this report. The information in these appendices was also combined into a single, separate document *The WRITE-LEARNER Computer-Assisted Instruction System User's Guide*. As mentioned previouslly, a distribution plan for this document had not been developed as of submission of this report.

V. Recommendations

As with many programming endeavors, one of the most difficult decisions of this research project was knowing when to stop. Any computer program can be improved and enhanced virtually without end. Therefore, this programming endeavor stopped when the author was satisfied with the results and would be willing to use the system. The program can be used as it stands or can be improved to the user's content.

In retrospect, many changes would have been made by the author had time permitted. However, these changes will have to wait for future versions of the program. It is the sincerest desire of this researcher that the program grow and expand as users improve and change the program. Nevertheless, recommendations and recommendations for further study are provided.

Recommendations

Recommendation Number One. Recommend that the Air Force Institute of Technology (AFIT) investigate the use of this computer-assisted instruction system for use in exportable education. Current policies include sending field training teams to Air Force bases for training. Many courses could be developed and taught using this computer-assisted instruction system, thereby saving the expense of sending training teams to remote locations.

Recommendation Number Two. Recommend that components of the Department of Defense make this computer-assisted instruction system available to operating components for use. Recommend further that these agencies become repositories for enhancements and improvements to the programs as provided by using activities.

Recommendations for Further Research

Recommendation Number One. Extensive field testing of the final product of this research project was not possible. This testing, comparing the relative effectiveness of this computer-assisted instruction system to other training methods (classroom, programmed instruction, self-study), could be performed.

Recommendation Number Two. Many enhancements and improvements to this computer-assisted instruction system are possible as a research project. Improvements to the editor portion of WRITE/BAS are possible. Computer graphic capabilities could be added.

Recommendation Number Three. The potential exists for linking this computer-assisted instruction system to multimedia presentations (video disk, photographic slides, and simulation equipment), especially using the TARGA 16 AT&T system.

Recommendation Number Four. The programs could be converted to operate on different microcomputers or converted to operate on a timesharing, mainframe computer.

Appendix A: LEARNER/BAS Program Documentation

Contents

LEARNER/BAS Program Overview

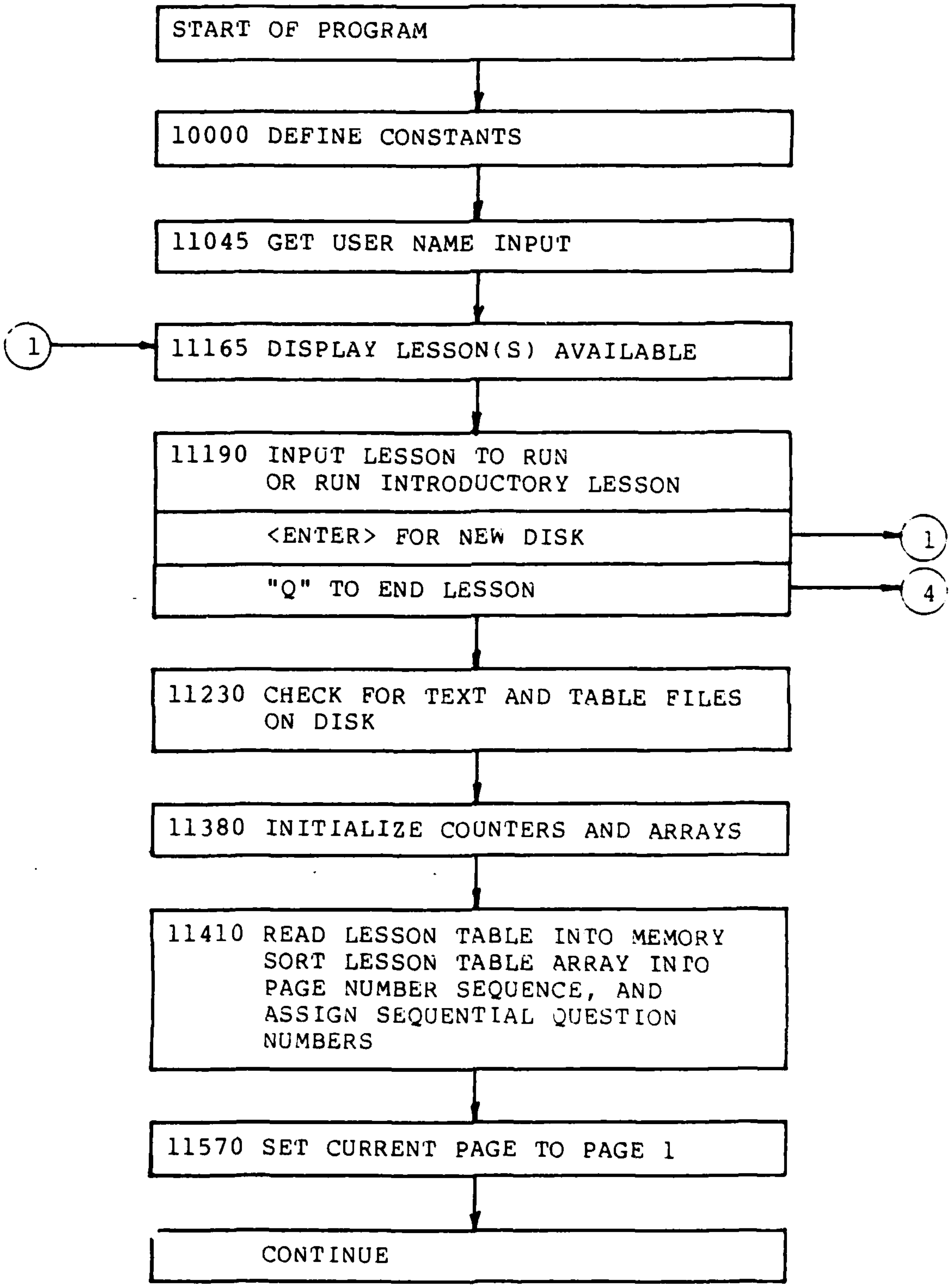
LEARNER/BAS is a program written in the BASIC programming language which is one component of the WRITE-LEARNER computer-assisted instruction system. The system was developed on a Tandy/Radio Shack Model IV microcomputer and subsequently converted and tested on a Zenith Z-248 (IBM PC/AT compatible) microcomputer. LEARNER/BAS administers interactive, computer-assisted instruction courseware modules developed using the WRITE/BAS program.

This appendix contains the documentation for the LEARNER/BAS program. Included are comments about program operation; block diagrams to illustrate the program logic; a description of the variables used in the program; cross-reference listings by variable name, line number, and BASIC keywords; and the program listings (TRSDOS and MSDOS versions). A user's guide for the program is included as a separate appendix.

LEARNER/BAS is essentially a specialized data-base manager. Lesson courseware is stored in formatted, direct-access, disk records which are read as required during lesson execution. A header record contains certain data about a lesson page (page type, page number, and jump pages). Following the header record are from 1 to 20 text records which comprise the lesson page. A second file, a sequential ASCII lesson table file, is generated from the lesson text file and is read into memory during lesson execution to allow improved response times in locating records in the text file. A third file, a sequential ASCII student file, is generated during lesson execution and stores student data (student name, date lesson executed, and total/correct/incorrect question responses) about the lesson use. Refer to the <u>LEARNER/BAS Record File Formats</u> for a description of these files.

Figure A.1 is a simplified block diagram illustrating the operation of LEARNER/BAS. Line numbers refer to both versions of the program (TRSDOS and MSDOS) since line number consistency was maintained during program conversion.

If attempting to compile this program or convert it to operate on another type of computer, several cautions are in order. First, the program uses REMARK line numbers in program flow. GOTO and GOSUB commands must be changed to reflect deletions of REMARK statements. Second, the MSDOS version of LEARNER/BAS emulates certain functions of the TRSDOS version (PRINT CHR$(30) in TRSDOS BASIC erases display to end of current line which was emulated in MSDOS BASIC by LOCATE xx,,yy:PRINT STRINGS$(81-POS(0),32):LOCATE xx,yy). Since the TRSDOS version of this program was the original version written, any conversion should begin with that program listing and the cross-reference listings included in this appendix.


START OF PROGRAM
10000 DEFINE CONSTANTS
11045 GET USER NAME INPUT
1
11165 DISPLAY LESSON(S) AVAILABLE
11190 INPUT LESSON TO RUN OR RUN INTRODUCTORY LESSON
<ENTER> FOR NEW DISK
1
"Q" TO END LESSON
4
11230 CHECK FOR TEXT AND TABLE FILES ON DISK
11380 INITIALIZE COUNTERS AND ARRAYS
11410 READ LESSON TABLE INTO MEMORY SORT LESSON TABLE ARRAY INTO PAGE NUMBER SEQUENCE, AND ASSIGN SEQUENTIAL QUESTION NUMBERS
11570 SET CURRENT PAGE TO PAGE 1
CONTINUE

NOTE: → (n) indicates branch to corresponding (n) →.

Figure A.1: LEARNER/BAS Block Diagram

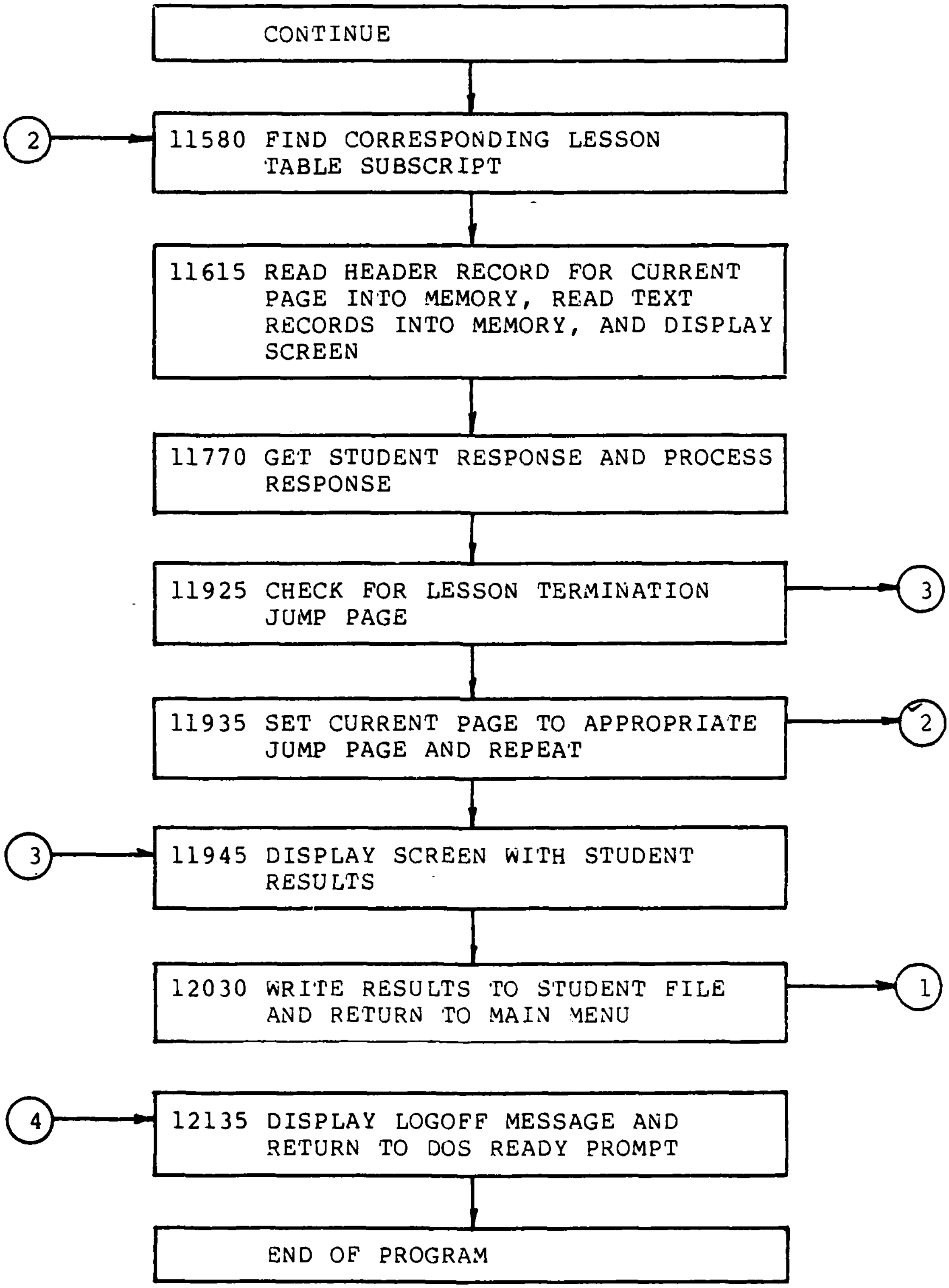

CONTINUE
2
11580 FIND CORRESPONDING LESSON TABLE SUBSCRIPT
11615 READ HEADER RECORD FOR CURRENT PAGE INTO MEMORY, READ TEXT RECORDS INTO MEMORY, AND DISPLAY SCREEN
11770 GET STUDENT RESPONSE AND PROCESS RESPONSE
11925 CHECK FOR LESSON TERMINATION JUMP PAGE
3
11935 SET CURRENT PAGE TO APPROPRIATE JUMP PAGE AND REPEAT
2
3
11945 DISPLAY SCREEN WITH STUDENT RESULTS
12030 WRITE RESULTS TO STUDENT FILE AND RETURN TO MAIN MENU
1
4
12135 DISPLAY LOGOFF MESSAGE AND RETURN TO DOS READY PROMPT
END OF PROGRAM

Figure A.1: LEARNER/BAS Block Diagram (continued)

LEARNER/BAS Variable List

A$	String for user input from keyboard.
AJUMP%	Go to lesson page number for <A>/<T>/<ENTER> response.
ANSWER$	Correct answer for question (A-E or T/F).
BJUMP%	Go to lesson page number for <B>/<F> response.
BUFi$	Temporary buffer variable for use with direct access files (i=integer value).
CJUMP%	Go to lesson page number for <C> response.
COUNT%	Counter for finding array subscript in lesson table.
DASH$	String of 80 dash characters.
DJUMP%	Go to lesson page number for <D> response.
DUMMY$	Dummy variable for reading unused portions of direct access files.
EJUMP%	Go to lesson page number for <E> response.
ENTER$	"Press <ENTER>..." instruction line.
FLAG%	Flag variable returned from subroutines to identify specific disk errors.
GRADE%	Student's grade for lesson answers (100 point scale).
I	FOR/NEXT loop counter.
MORE%	Number of associated disk records required to generate lesson page in lesson text file.
MORE%(	Number of associated disk records required to generate lesson page in lesson table array.
NUMPAGES%	Number of pages in lesson text file.
PAGE%(	Lesson page number in lesson table array.
PAGE%	Lesson page number in lesson text file.
PROMPT$	"Press <letter> of your choice..." instruction line.
QDATA$	"Enter requested data and press <ENTER>..." instruction line.
QINST$	Instruction line to be displayed on lesson screen page.
QMAT%	Array subscript in lesson table.
QNAME$	Lesson name to be displayed on lesson screen page.
QNUM%(	Question number in lesson table.

QNUM%	Temporary counter for assigning question numbers in lesson table.
QPAGE$	Page number to be displayed on lesson screen page.
QPAGE%	Lesson page number being displayed in lesson execution/active page number.
QTEST$	File name for determining if file is on disk.
QUEST$	Question number to be displayed on lesson screen page.
RANSWER$	"Right response! Press <ENTER>..." instruction line.
RCOUNT%	Number of correct responses to questions asked.
START%(	Starting lesson text file record number in lesson table.
START%	Starting lesson text file record number in lesson table file.
STUDENT$	Student's name (first initial and last name, no space).
STUFILE$	Name of file for writing student lesson data.
TABLE$	File name of lesson table disk file.
TCOUNT%	Total number of questions asked.
TEST%	Temporary variable for testing capital character input during student name input.
TEXT$(	Temporary array holding text lines of lesson pages.
TEXT$	Disk file name for lesson text file.
TNAME$	Name of lesson to run.
TYPE$	Lesson page type indicator (#=text page; ?=question page).
TYPE$(	Lesson page type indicator in lesson table.
WANSWER$	"Wrong response! Press <ENTER>..." instruction line.
WCOUNT%	Number of incorrect responses to questions asked.
WQUEST$(	Array holding incorrect responses to questions asked.
WQUEST%(	Array holding question numbers to which student responded incorrectly.
YNINST$	"Press <Y>es or <N>o to continue..." instruction line.

LEARNER/BAS Record File Formats

LEARNER/BAS uses three files produced by the WRITE/BAS program and produces one file for use by the WRITE/BAS program. The following pages contain the record file formats for these files. Although sequential ASCII files are not formatted (data elements are variable length), lengths are provided for these data elements for reference purposes.

The first file used by LEARNER/BAS, <Lesson Name>/TXT is comprised of two different types of formatted, direct-access records which form a page "block". A header record containing information about the lesson page is followed by from 1 to 20 text records which contain the page text material.

The second file, <Lesson Name>/TAB, is a sequential ASCII file which contains certain information from the text header records. This file is loaded into memory permitting very rapid access of the proper text records.

The third file, <Lesson Name>/STU, is a sequential ASCII file which contains data generated during lesson use. This file is appended on completion of each execution of a lesson module. If the file does not exist, then it is created by the LEARNER/BAS program.

File: <Lesson Name>/TXT (Header records)
Type: Formatted, direct access

Length	Variable/s	Type	Remarks
1	TYPE$ BUF1$	A	Page type; #=text page, ?=question page, null/blank= text record, *=deleted record.
5	PAGE% BUF2$	N	User defined page number; used for lesson branching/reference purposes.
5	AJUMP% BUF3$	N	Branch to page for <A> response to multiple choice question, <T> response to true/false question, or <ENTER> key for text page.
5	BJUMP% BUF4$	N	Branch to page for <B> response to multiple choice question or <F> response to true/false question.
5	CJUMP% BUF5$	N	Branch to page for <C> response to multiple choice question.
5	DJUMP% BUF6$	N	Branch to page for <D> response to multiple choice question.
5	EJUMP% BUF7$	N	Branch to page for <E> response to multiple choice question.
1	ANSWER$ BUF8$	A	Correct response for question; must equal A-E or T/F.
3	MORE% BUF9$	N	Number of additional records of page (i.e., number of text lines); must equal 1-20 (third character reserved by BASIC for sign).
46	DUMMY$		Not used in header records.

Total record length = 81

File: <Lesson Name>/TXT (text records)
Type: Formatted, direct access

Length	Variable/s	Type	Remarks
1	-	A	Null/blank=text record; *=deleted record.
80	TEXT$	A/N	Text line.

Total record length = 81

File: <Lesson Name>/TAB
Type: Sequential ASCII

Length	Variable/s	Type	Remarks
5	PAGE% BUF2$	N	User defined page number; used for lesson branching.
4	START%	N	Starting file record number for PAGE% header record in <Lesson Name>/TXT file.
4	MORE%	N	Number of text records in <Lesson Name>/TXT file comprising PAGE%.
1	TYPE$	A	Page type.

File: <Lesson Name>/STU
Type: Sequential ASCII

Length	Variable/s	Type	Remarks
NA	STUDENT$	A	Student name (first initial and last name; no space).
NA	DATE$	A/N	Variable length; format as returned by applicable BASIC/DOS.
4	TCOUNT%	N	Total questions asked during lesson (may differ from total questions in a lesson due to branching and/or repeated questions).
4	RCOUNT%	N	Number of correct responses.
4	WCOUNT%	N	Number of incorrect responses.
4	WQUEST%(i)	N	Question number answered incorrectly (i=1 to WCOUNT%).
1	WQUEST$(i)	A	Incorrect response to question (i=1 to WCOUNT).

LEARNER/BAS BASIC Variables Cross-Reference List

```
A$
          215    220    225    245    250    255    275
          280    285    305    310    315    355  11145
        11830  11835  11840  11845  11850  11855  11860
        11890  11895  11900  11905  12240  12240
AJUMP%
        11630  11805  11830  11890
ANSWER$
        11655  11780  11785  11855  11860  11900  11905
BJUMP%
        11635  11835  11895
BUF
        11560  11560  11560  11560  11560  11560  11560
        11560  11560  11565  11565  11625  11630  11635
        11640  11645  11650  11655  11660  11685  11690
CJUMP%
        11640  11840
COUNT%
        11585  11590  11595  11595  11605
DASH$
          130    135  10020
DJUMP%
        11645  11845
DUMMY$
        11560  11685
EJUMP%
        11650  11850
ENTER$
        10025  11250  11330  11710  11950
FLAG%
          155    175    180    185    190  11245  11255
        11325  12080  12085
GRADE%
        11955  12015
I
        11430  11440  11440  11440  11440  11445  11445
        11460  11470  11475  11480  11485  11490  11495
        11500  11510  11520  11525  11530  11540  11675
        11680  11690  11695  11745  11750  11755  12095
        12100  12100  12105  12180  12180  12210  12210
        12210  12210
J
        11475  11480  11485  11490  11495  11500  11505
MORE%
        11390  11660  11675  11745
MORE%(
        11030  11400  11440  11495  11495
NUMPAGES%
        11460  11470  11475  11520
```

```
PAGE%(
          11030  11400  11440  11480  11480  11485  11485
          11590
PAGE%
          11390
PROMPT$
          10030  11715
QDATA$
          10040  11050  11170
QINST$
            135  11050  11120  11170  11250  11330  11415
          11710  11715  11950  12040  12145
QMAT%
            350  11605  11620  11680  11735
QNAME$
            130  11050  11120  11170  11250  11330  11415
          11725  11950  12040  12145
QNUM%(
            350  11035  11405  11530  11735
QNUM%
          11395  11515  11530  11535  11535
QPAGE$
            130  11050  11120  11170  11250  11330  11415
          11720  11950  12040  12145
QPAGE%
          11570  11590  11720  11805  11830  11835  11840
          11845  11850  11890  11895  11930
QTEST$
            165  11240  11320  12070
QUEST$
            135  11170  11735  11735  11950  12040  12145
RANSWER$
            375  10045
RCOUNT%
            385    385  11385  11955  11985  12090
STAR$
          12200  12210  12210
START%(
          11030  11400  11440  11490  11490  11620  11680
START%
          11390
STUDENT$
          11075  11090  12090
STUFILE$
          12065  12070  12080  12085
TABLE$
          11220  11320  11425
TCOUNT%
          11385  11730  11730  11955  11965  12090
TEST%
          11090  11095  11095
TEXT$(
          11035  11405  11690  11750
```

```
TEXT$
            11215   11240   11395   11555
TNAME$
            11150   11190   11195   11200   11205   11210   11210
            11210   11215   11220   11260   11275   11415   11725
            11950   12035   12065
TYPE$
            11390   11625   11710   11715   11730   11735   11775
TYPE$(
            11030   11400   11440   11500   11500   11525
WANSWER$
              335   10050
WCOUNT%
              345     345     350     355   11385   11995   12090
            12095
WQUEST$(
              355   11035   11405   12100
WQUEST$
            11395
WQUEST%(
              350   11035   11405   12100
WQUEST%
            11395
YNINST$
            10035   11120
```

LEARNER/BAS Line Number Cross-Reference List

```
  125 => 11050  11120  11170  11250  11330  11415  11740
         11950  12040  12145
  150 => 11240  11320  12075
  175 =>   160
  195 =>   175    180    185
  200 =>   170    195
  210 => 11140
  215 =>   225
  240 => 11825
  245 =>   255
  270 => 11885
  275 =>   285
  300 => 11295  11365  11800  11865  11910  12020
  305 =>   315
  330 => 11855  11900
  370 => 11860  11905
10000 =>    10
11045 => 12245
11070 => 11105
11115 => 11095
11165 => 11145  11195  11300  11370  12035  12125
11215 => 11155  11205
11260 => 11255
11265 => 11255
11275 => 11255
11280 => 11255
11290 => 11260  11270  11275  11285
11310 => 11245
11380 => 11325
11435 => 11450
11455 => 11435
11485 => 11480
11505 => 11480
11540 => 11525
11580 => 11935
11590 => 11600
11605 => 11590
11795 => 11775
11820 => 11780
11880 => 11785
11925 => 11810  11870  11915
11945 => 11930
12145 =>   220    250    280    310  11200
12195 =>   200  11025
12240 => 12240
```

LEARNER/BAS BASIC Keywords Cross-Reference List

```
*          11955
+            345     385   10025   10030   10035   10040   10045
           10045   10050   10050   11215   11220   11415   11445
           11475   11535   11595   11680   11730   11735   12040
           12065   12200
-          11210   11460   11470   11955
<            315   11095   11525   11780   11785   11855   11900
           12240
=            155     175     175     180     180     185     185
             190     215     220     225     245     250     255
             275     280     285     305     310     345     350
             355     385   10020   10025   10030   10035   10040
           10045   10050   11050   11050   11050   11090   11095
           11095   11120   11120   11120   11145   11150   11170
           11170   11170   11170   11195   11200   11205   11210
           11215   11220   11240   11245   11250   11250   11250
           11320   11325   11330   11330   11330   11385   11385
           11385   11415   11415   11415   11430   11445   11460
           11470   11475   11515   11520   11530   11535   11570
           11585   11590   11595   11605   11625   11630   11635
           11640   11645   11650   11655   11660   11675   11685
           11690   11710   11710   11715   11715   11720   11725
           11730   11730   11735   11735   11735   11745   11775
           11805   11830   11830   11835   11835   11840   11840
           11845   11845   11850   11850   11860   11890   11890
           11895   11895   11905   11930   11950   11950   11950
           11950   11955   12035   12040   12040   12040   12040
           12065   12070   12080   12085   12095   12145   12145
           12145   12145   12180   12200   12210   12240
>            315   11095   11480   11525   11780   11785   11855
           11900   12240
AND        11095
ASC        11090
CHR$         220     250     280     310     315   12240
CLOSE        200   11455   12035   12110   12140   12185   12245
CLS          130   12205
DATE$      12090
DIM        11030   11035   11400   11405
ELSE       11480   11735
END        12185   12250
EOF        11435
ERASE      11390   11395
ERL        12220
ERR          175     180     185   12220
ERROR        160     200   11025
FIELD      11560   11565
FOR        11470   11475   11520   11675   11745   12095   12180
           12210
GET        11620   11680
GOSUB      11050   11120   11140   11170   11240   11250   11295
           11320   11330   11365   11415   11740   11800   11825
```

```
GOSUB     11855  11860  11865  11885  11900  11905  11910
          11950  12020  12040  12075  12145
GOTO         10    160    170    175    180    185    200
          11025  11105  11155  11255  11260  11270  11275
          11285  11300  11370  11450  11600  11810  11870
          11915  11935  12125
IF          175    180    185    220    225    250    255
            280    285    310    315  11095  11145  11195
          11200  11205  11245  11325  11435  11480  11525
          11590  11710  11715  11730  11735  11775  11780
          11785  11830  11835  11840  11845  11850  11855
          11860  11890  11895  11900  11905  11930  12035
          12080  12085  12240
INKEY$    12240
INPUT       215    245    275    305  11075  11190  11440
INSTR       225    255    285  11205  11210  11780  11785
LEFT$     11090  11210
NEXT      11505  11510  11540  11695  11755  12105  12180
          12210
ON          160    200  11025  11255
OPEN        165  11425  11555  12080  12085
OPTION    11025
PRINT       130    130    130    135    135    135    135
            335    340    375    380  11055  11060  11065
          11070  11100  11100  11100  11125  11130  11135
          11175  11180  11260  11265  11270  11275  11280
          11285  11290  11335  11340  11345  11345  11350
          11355  11355  11360  11420  11750  11760  11960
          11965  11970  11975  11980  11985  11990  11995
          12000  12005  12010  12015  12015  12045  12150
          12155  12160  12165  12170  12175  12210  12210
          12210  12210  12215  12220  12225  12230  12235
REM           1      2      3      4      5      6      7
              8      9     10     15    100    105    110
            115    120    125    145    150    175    180
            185    190    205    210    235    240    265
            270    295    300    325    330    365    370
            395  10000  10005  10010  10015  10055  11000
          11005  11010  11015  11020  11040  11045  11080
          11085  11110  11115  11150  11160  11165  11195
          11225  11230  11235  11305  11310  11315  11375
          11380  11410  11465  11545  11550  11570  11575
          11580  11610  11615  11665  11670  11700  11705
          11765  11770  11775  11780  11785  11790  11795
          11815  11820  11875  11880  11920  11925  11935
          11940  11945  12025  12030  12050  12055  12060
          12115  12120  12130  12135  12180  12190  12195
RESUME      195  12245
RETURN      140    200    230    260    290    320    360
            390
SOUND       340
STR$      11720  11735
```

```
STRING$   10020  10025  10030  10035  10040  10045  10050
          11415  12040  12200  12200
SWAP      11485  11490  11495  11500
SYSTEM    11185  12185
THEN        175    180    185    220    225    250    255
            280    285    310    315  11095  11145  11195
          11200  11205  11245  11325  11435  11480  11525
          11590  11710  11715  11730  11735  11775  11780
          11785  11830  11835  11840  11845  11850  11855
          11860  11890  11895  11900  11905  11930  12035
          12080  12085  12240
TO        11470  11475  11520  11675  11745  12095  12180
          12210
USING     11965  11985  11995  12015
VAL       11630  11635  11640  11645  11650  11660
WRITE     12090  12100
```

LEARNER/BAS Program Listing (TRSDOS Version)

```
1 '*********************************************************
2 '* LEARNER/BAS - COMPUTER-ASSISTED INSTRUCTION SOFTWARE  *
3 '* ROBERT MASON, LT, SC, USN                             *
4 '* AIR FORCE INSTITUTE OF TECHNOLOGY                     *
5 '* SCHOOL OF SYSTEMS AND LOGISTICS                       *
6 '* MAY 1987                                              *
7 '* TANDY/RADIO SHACK MODEL IV VERSION 01.00.00.          *
8 '*********************************************************
9 '
10 GOTO 10000   'JUMP TO START OF MAIN PROGRAM
15 '
100 '*********************************************
105 '*              SUBROUTINES                  *
110 '*           (LINES 100-9999)               *
115 '*********************************************
120 '
125 '   SUBROUTINE - PRINT SCREEN BOILERPLATE
130 CLS
          :PRINT@(0,0),QNAME$;
          :PRINT@(0,76),QPAGE$;
          :PRINT@(1,0),DASH$;
135 PRINT@(22,0),DASH$;
          :PRINT@(23,0),QINST$;
          :PRINT@(0,31),QUEST$ ;
          :PRINT@(2,0),;
140 RETURN
145 '
150 '   SUBROUTINE - CHECK FOR FILENAME ON DISK
155 FLAG%=0
160 ON ERROR GOTO 175
165 OPEN "I",1,QTEST$
170 GOTO 200
175 IF ERR=53 THEN FLAG%=1
          :GOTO 195   '   FILE NOT FOUND ERROR
180 IF ERR=57 THEN FLAG%=2
          :GOTO 195   '   DEVICE I/O ERROR
185 IF ERR=64 THEN FLAG%=3
          :GOTO 195   '   BAD FILE NAME ERROR
190 FLAG%=4                  '   UNKNOWN ERROR
195 RESUME 200
200 CLOSE
          :ON ERROR GOTO 12195
          :RETURN
205 '
210 '   SUBROUTINE - WAIT FOR Y/N INPUT
215 A$=INPUT$(1)
220 IF A$=CHR$(5) THEN 12145
225 IF INSTR("YN",A$)=0 THEN 215
230 RETURN
235 '
```

```
240 '   SUBROUTINE - WAIT FOR A/B/C/D/E INPUT
245 A$=INPUT$(1)
250 IF A$=CHR$(5) THEN 12145
255 IF INSTR("ABCDE",A$)=0 THEN 245
260 RETURN
265 '
270 '   SUBROUTINE - WAIT FOR T/F INPUT
275 A$=INPUT$(1)
280 IF A$=CHR$(5) THEN 12145
285 IF INSTR("TF",A$)=0 THEN 275
290 RETURN
295 '
300 '   SUBROUTINE - WAIT FOR <ENTER> INPUT
305 A$=INPUT$(1)
310 IF A$=CHR$(5) THEN 12145
315 IF A$<>CHR$(13) THEN 305
320 RETURN
325 '
330 '   SUBROUTINE - HANDLE INCORRECT RESPONSE
335 PRINT@(23,1),WANSWER$;
340 PRINT@(23,0),;:SOUND 5,1
345 WCOUNT%=WCOUNT%+1
350 WQUEST%(WCOUNT%)=QNUM%(QMAT%)
355 WQUEST$(WCOUNT%)=A$
360 RETURN
365 '
370 '   SUBROUTINE - HANDLE CORRECT RESPONSE
375 PRINT@(23,1),RANSWER$;
380 PRINT@(23,0),;
385 RCOUNT%=RCOUNT%+1
390 RETURN
395 '
10000 '*********************************************
10005 '*   CONSTANT TABLE AND DEFINED FUNCTIONS    *
10010 '*       (LINES 10000-10999)                 *
10015 '*********************************************
10020 DASH$=STRING$(80,45)
10025 ENTER$=STRING$(22,32)+"Press <ENTER> to continue."
10030 PROMPT$=STRING$(22,32)+"Press <letter> of your choice.
      "
10035 YNINST$=STRING$(23,32)+"Press <Y>es or <N>o to continu
      e."
10040 QDATA$=STRING$(14,32)+"Enter requested data and press
      <ENTER> to continue."
10045 RANSWER$="Right response!"+STRING$(10,32)+"Press <ENTE
      R> to continue."
10050 WANSWER$="Wrong response!"+STRING$(10,32)+"Press <ENTE
      R> to continue."
10055 '
11000 '*********************************************
11005 '*             MAIN PROGRAM                  *
11010 '*          (LINES 11000-39999)              *
11015 '*********************************************
```

```
11020 '
11025 OPTION BASE 1
        :ON ERROR GOTO 12195
11030 DIM PAGE%(200),START%(200),MORE%(200),TYPE$(200)
11035 DIM TEXT$(20),WQUEST%(200),WQUEST$(200),QNUM%(200)
11040 '
11045 '   PRINT SCREEN AND GET USER NAME
11050 QNAME$="MAIN MENU"
        :QINST$=QDATA$
        :QPAGE$=""
        :GOSUB 125
11055 PRINT"ENSURE THAT THE CAPS LOCK KEY IS DEPRESSED AND R
      EMAINS"
11060 PRINT"DEPRESSED FOR THE DURATION OF THIS LESSON!"
11065 PRINT
11070 PRINT"Input your first initial and last name in capita
      l letters with"
11075 INPUT"no space (example: RSMITH):  ",STUDENT$
11080 '
11085 '   CHECK FOR CAPITAL INPUT - REPEAT IF NECESSARY
11090 TEST%=ASC(LEFT$(STUDENT$,1))
11095 IF TEST%>=65 AND TEST%<=90 THEN 11115
11100 PRINT
        :PRINT"Erroneous input...ensure that the caps lock
         key is depressed!"
        :PRINT
11105 GOTO 11070
11110 '
11115 '   OFFER TO RUN INTRODUCTORY LESSON
11120 QNAME$="MAIN MENU"
        :QINST$=YNINST$
        :QPAGE$=""
        :GOSUB 125
11125 PRINT@(2,0),"Do you want to run a short introductory l
      esson about this"
11130 PRINT"computer-assisted instruction program before sta
      rting a lesson?"
11135 PRINT@(23,0),;
11140 GOSUB 210
11145 IF A$="N" THEN 11165
11150 TNAME$="INTRO"   '   RUN INTRO LESSON
11155 GOTO 11215
11160 '
11165 '   PRINT SCREEN, LESSON CATALOGUE, AND GET LESSON
        FILE NAME
11170 QNAME$="MAIN MENU"
        :QINST$=QDATA$
        :QPAGE$=""
        :QUEST$=""
        :GOSUB 125
11175 PRINT@(2,0),"The following lesson files are available
      on this disk:"
11180 PRINT
```

```
11185 SYSTEM "DIR $/TXT:1 (ALL=OFF)"
11190 INPUT"Enter lesson to run or change data disk and pres
      s <ENTER>: ",TNAME$
11195 IF TNAME$="" THEN 11165   'NEW DISK CATALOG
11200 IF TNAME$="Q" THEN 12145
11205 IF INSTR(TNAME$,"/")=0 THEN 11215
11210 TNAME$=LEFT$(TNAME$,(INSTR(TNAME$,"/")-1))
11215 TEXT$=TNAME$+"/TXT"
11220 TABLE$=TNAME$+"/TAB"
11225 '
11230 '   CHECK FOR LESSON TEXT FILE ON DISK, PRINT ERROR
          MESSAGE, AND
11235 '   RETURN TO MAIN MENU IF NECESSARY
11240 QTEST$=TEXT$
          :GOSUB 150
11245 IF FLAG%=0 THEN 11310
11250 QNAME$="ERROR"
          :QINST$=ENTER$
          :QPAGE$=""
          :GOSUB 125
11255 ON FLAG% GOTO 11260,11265,11275,11280
11260 PRINT TNAME$;" is not on these disks."
          :GOTO 11290
11265 PRINT"A device input/output error has occurred!"
11270 PRINT"Terminate this session and contact your training
      supervisor."
          :GOTO 11290
11275 PRINT TNAME$;" is not a valid lesson name."
          :GOTO 11290
11280 PRINT"An unknown disk error has occurred!"
11285 PRINT"Terminate this session and contact your training
      supervisor."
          :GOTO 11290
11290 PRINT@(23,0),;
11295 GOSUB 300
11300 GOTO 11165
11305 '
11310 '   CHECK FOR LESSON TABLE FILE ON DISK, PRINT ERROR
          MESSAGE, AND
11315 '   RETURN TO MAIN MENU IF REQUIRED
11320 QTEST$=TABLE$:GOSUB 150
11325 IF FLAG%=0 THEN 11380
11330 QNAME$="ERROR"
          :QINST$=ENTER$
          :QPAGE$=""
          :GOSUB 125
11335 PRINT"Unable to run this lesson...the corresponding le
      sson table"
11340 PRINT"file is not on these disks."
11345 PRINT
          :PRINT"Contact your training supervisor to make co
           rrections to"
11350 PRINT"this lesson disk."
```

```
11355 PRINT
        :PRINT"You may try another lesson or terminate thi
         s session."
11360 PRINT@(23,0),;
11365 GOSUB 300
11370 GOTO 11165
11375 '
11380 '   INITIALIZE QUESTION COUNTERS AND ARRAYS
11385 TCOUNT%=0
        :RCOUNT%=0
        :WCOUNT%=0
11390 ERASE PAGE%,START%,MORE%,TYPE$
11395 ERASE TEXT$,WQUEST%,WQUEST$,QNUM%
11400 DIM PAGE%(200),START%(200),MORE%(200),TYPE$(200)
11405 DIM TEXT$(20),WQUEST%(200),WQUEST$(200),QNUM%(200)
11410 '   READ LESSON TABLE
11415 QNAME$=TNAME$
        :QINST$=STRING$(20,32)+"Loading lesson table...ple
         ase wait..."
        :QPAGE$=""
        :GOSUB 125
11420 PRINT@(23,0),;
11425 OPEN "I",1,TABLE$
11430 I=1
11435 IF EOF(1) THEN 11455
11440 INPUT#1,PAGE%(I),START%(I),MORE%(I),TYPE$(I)
11445 I=I+1
11450 GOTO 11435
11455 CLOSE 1
11460 NUMPAGES%=I-1
11465 '   SORT TABLE INTO PAGE # SEQUENCE
11470 FOR I=1 TO NUMPAGES%-1
11475 FOR J=I+1 TO NUMPAGES%
11480 IF PAGE%(I)>PAGE%(J) THEN 11485 ELSE 11505
11485 SWAP PAGE%(I),PAGE%(J)
11490 SWAP START%(I),START%(J)
11495 SWAP MORE%(I),MORE%(J)
11500 SWAP TYPE$(I),TYPE$(J)
11505 NEXT J
11510 NEXT I
11515 QNUM%=1
11520 FOR I=1 TO NUMPAGES%
11525 IF TYPE$(I)<>"?" THEN 11540
11530 QNUM%(I)=QNUM%
11535 QNUM%=QNUM%+1
11540 NEXT I
11545 '
11550 '   OPEN LESSON BUFFER AND START LESSON WITH PAGE 1
11555 OPEN "D",2,TEXT$,81
11560 FIELD 2,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5
      AS BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,4
      6 AS DUMMY$
11565 FIELD 2,1 AS BUF10$,80 AS BUF11$
```

```
11570 QPAGE%=1   'SET COUNTER FOR FIRST PAGE OF LESSON
11575 '
11580 '   FIND CORRESPONDING TABLE SUBSCRIPT
11585 COUNT%=1
11590 IF PAGE%(COUNT%)=QPAGE% THEN 11605
11595 COUNT%=COUNT%+1
11600 GOTO 11590
11605 QMAT%=COUNT%
11610 '
11615 '   READ DATA FOR A PAGE
11620 GET 2,START%(QMAT%)
11625 TYPE$=BUF1$
11630 AJUMP%=VAL(BUF3$)
11635 BJUMP%=VAL(BUF4$)
11640 CJUMP%=VAL(BUF5$)
11645 DJUMP%=VAL(BUF6$)
11650 EJUMP%=VAL(BUF7$)
11655 ANSWER$=BUF8$
11660 MORE%=VAL(BUF9$)
11665 '
11670 '   READ THE TEXT RECORDS FOR THIS PAGE
11675 FOR I=1 TO MORE%
11680 GET 2,(START%(QMAT%)+I)
11685 DUMMY$=BUF10$
11690 TEXT$(I)=BUF11$
11695 NEXT I
11700 '
11705 '   DECIDE ON PROPER INSTRUCTION LINE AND DISPLAY PAGE
11710 IF TYPE$="#" THEN QINST$=ENTER$
11715 IF TYPE$="?" THEN QINST$=PROMPT$
11720 QPAGE$=STR$(QPAGE%)
11725 QNAME$=TNAME$
11730 IF TYPE$="?" THEN TCOUNT%=TCOUNT%+1
11735 IF TYPE$="?" THEN QUEST$="QUESTION #"+STR$(QNUM%(QMAT%
      )) ELSE QUEST$=""
11740 GOSUB 125
11745 FOR I=1 TO MORE%
11750 PRINT TEXT$(I);
11755 NEXT I
11760 PRINT@(23,0),;
11765 '
11770 '   ROUTE TO PROPER LINE FOR HANDLING RESPONSE TO PAGE
          TYPE
11775 IF TYPE$="#" THEN 11795   'TEXT PAGE
11780 IF INSTR("ABCDE",ANSWER$)<>0 THEN 11820   'MULTIPLE
      CHOICE QUESTION
11785 IF INSTR("TF",ANSWER$)<>0 THEN 11880   'TRUE/FALSE
      QUESTION
11790 '
11795 '   HANDLE TEXT PAGE USER RESPONSE
11800 GOSUB 300
11805 QPAGE%=AJUMP%
11810 GOTO 11925
```

```
11815 '
11820 '   HANDLE MULTIPLE CHOICE QUESTION RESPONSE
11825 GOSUB 240
11830 IF A$="A" THEN QPAGE%=AJUMP%
11835 IF A$="B" THEN QPAGE%=BJUMP%
11840 IF A$="C" THEN QPAGE%=CJUMP%
11845 IF A$="D" THEN QPAGE%=DJUMP%
11850 IF A$="E" THEN QPAGE%=EJUMP%
11855 IF A$<>ANSWER$ THEN GOSUB 330
11860 IF A$=ANSWER$ THEN GOSUB 370
11865 GOSUB 300
11870 GOTO 11925
11875 '
11880 '   HANDLE TRUE/FALSE QUESTION RESPONSE
11885 GOSUB 270
11890 IF A$="T" THEN QPAGE%=AJUMP%
11895 IF A$="F" THEN QPAGE%=BJUMP%
11900 IF A$<>ANSWER$ THEN GOSUB 330
11905 IF A$=ANSWER$ THEN GOSUB 370
11910 GOSUB 300
11915 GOTO 11925
11920 '
11925 '   CHECK FOR BRANCH TO LAST PAGE
11930 IF QPAGE%=9999 THEN 11945
11935 GOTO 11580    '   REPEAT PROCESS FOR NEW PAGE
11940 '
11945 '   PRINT SCREEN WITH STUDENT RESULTS
11950 QNAME$=TNAME$:QINST$=ENTER$:QUEST$="":QPAGE$="":GOSUB
      125
11955 GRADE%=(RCOUNT%/TCOUNT%)*100
11960 PRINT"During this lesson you were asked ";
11965 PRINT USING "###";TCOUNT%;
11970 PRINT" questions."
11975 PRINT
11980 PRINT"You answered ";
11985 PRINT USING "###";RCOUNT%;
11990 PRINT" correctly and ";
11995 PRINT USING "###";WCOUNT%;
12000 PRINT" incorrectly."
12005 PRINT
12010 PRINT"Your grade for this lesson is ";
12015 PRINT USING "###.";GRADE%
        :PRINT@(23,0),;
12020 GOSUB 300
12025 '
12030 '   END LESSON ROUTINE - WRITE STUDENT FILE
12035 CLOSE
        :IF TNAME$="INTRO" THEN 11165
```

```
12040 QNAME$="ENDING LESSON"
        :QPAGE$=""
        :QINST$=STRING$(20,32)+"Saving lesson results...pl
         ease wait."
        :QUEST$=""
        :GOSUB 125
12045 PRINT@(23,0),;
12050 '
12055 '   CHECK FOR EXISTING STUDENT FILE ON DRIVE 1 - OPEN
          IN CORRECT MODE
12060 '   WRITE DATA TO FILE
12065 STUFILE$=TNAME$+"/STU:1"
12070 QTEST$=STUFILE$
12075 GOSUB 150
12080 IF FLAG%=0 THEN OPEN "E",1,STUFILE$
12085 IF FLAG%=1 THEN OPEN "O",1,STUFILE$
12090 WRITE# 1, STUDENT$,DATE$,TCOUNT%,RCOUNT%,WCOUNT%
12095 FOR I=1 TO WCOUNT%
12100 WRITE# 1,WQUEST%(I),WQUEST$(I)
12105 NEXT I
12110 CLOSE
12115 '
12120 '   RETURN TO MAIN MENU
12125 GOTO 11165
12130 '
12135 '   TERMINATE SESSION ROUTINE
12140 CLOSE
12145 QNAME$="GOODBYE"
        :QINST$=""
        :QPAGE$=""
        :QUEST$=""
        :GOSUB 125
12150 PRINT "Thank you for using the LEARNER computer-assist
      ed instruction"
12155 PRINT"system."
12160 PRINT
12165 PRINT"Wait for the DOS ready prompt before removing di
      sks or turning"
12170 PRINT"machine off!"
12175 PRINT@(23,0),;
12180 FOR I=1 TO 3000
        :NEXT I    '   DELAY LOOP TO DISPLAY LOGOFF MESSAGE
12185 CLOSE
        :SYSTEM
        :END
12190 '
12195 '   PROGRAM FATAL ERROR ROUTINE
12200 STAR$=STRING$(10,32)+STRING$(60,42)
12205 CLS
12210 PRINT@(3,0),STAR$:FOR I=4 TO 15
        :PRINT@(I,10),"**";:PRINT@(I,68),"**";
        :NEXT I
        :PRINT@(15,0),STAR$;
```

```
12215 PRINT@(5,21),"FATAL PROGRAM ERROR DURING EXECUTION";
12220 PRINT@(7,25),"Error code ";ERR;" in line ";ERL;
12225 PRINT@(10,15),"Copy above data and deliver to training
      supervisor!";
12230 PRINT@(13,23),"Press <ENTER> to restart program.";
12235 PRINT@(13,22),;
12240 A$=INKEY$
          :IF A$<>CHR$(13) THEN 12240
12245 CLOSE
          :RESUME 11045
12250 END
```

LEARNER/BAS Program Listing (MSDOS Version)

```
1 '**********************************************************
2 '* LEARNER/BAS - COMPUTER-ASSISTED INSTRUCTION SOFTWARE   *
3 '* ROBERT MASON, LT, SC, USN                              *
4 '* AIR FORCE INSTITUTE OF TECHNOLOGY                      *
5 '* SCHOOL OF SYSTEMS AND LOGISTICS                        *
6 '* MAY 1987                                              *
7 '* IBM/PC VERSION 01.00.00.                              *
8 '**********************************************************
9 '
10 GOTO 10000   'JUMP TO START OF MAIN PROGRAM
15 '
100 '****************************************
105 '*              SUBROUTINES             *
110 '*           (LINES 100-9999)           *
115 '****************************************
120 '
125 '   SUBROUTINE - PRINT SCREEN BOILERPLATE
130 CLS
          :LOCATE 1,1
          :PRINT QNAME$
          :LOCATE 1,77
          :PRINT QPAGE$
          :LOCATE 2,1
          :PRINT DASH$;
135 LOCATE 23,1
          :PRINT DASH$;
          :LOCATE 24,1
          :PRINT QINST$;
          :LOCATE 1,32
          :PRINT QUEST$;
          :LOCATE 3,1
140 RETURN
145 '
150 '   SUBROUTINE - CHECK FOR FILENAME ON DISK
155 FLAG%=0
160 ON ERROR GOTO 175
165 OPEN QTESTS$ FOR INPUT AS #1
170 GOTO 200
175 IF ERR=53 THEN FLAG%=1
          :GOTO 195   '   FILE NOT FOUND ERROR
180 IF ERR=57 THEN FLAG%=2
          :GOTO 195   '   DEVICE I/O ERROR
185 IF ERR=64 THEN FLAG%=3
          :GOTO 195   '   BAD FILE NAME ERROR
190 FLAG%=4                 '   UNKNOWN ERROR
195 RESUME 200
200 CLOSE
          :ON ERROR GOTO 12195:RETURN
205 '
210 '   SUBROUTINE - WAIT FOR Y/N INPUT
```

```
215 A$=INPUT$(1)
220 IF A$=CHR$(5) THEN 12145
225 IF INSTR("YN",A$)=0 THEN 215
230 RETURN
235 '
240 '   SUBROUTINE - WAIT FOR A/B/C/D/E INPUT
245 A$=INPUT$(1)
250 IF A$=CHR$(5) THEN 12145
255 IF INSTR("ABCDE",A$)=0 THEN 245
260 RETURN
265 '
270 '   SUBROUTINE - WAIT FOR T/F INPUT
275 A$=INPUT$(1)
280 IF A$=CHR$(5) THEN 12145
285 IF INSTR("TF",A$)=0 THEN 275
290 RETURN
295 '
300 '   SUBROUTINE - WAIT FOR <ENTER> INPUT
305 A$=INPUT$(1)
310 IF A$=CHR$(5) THEN 12145
315 IF A$<>CHR$(13) THEN 305
320 RETURN
325 '
330 '   SUBROUTINE - HANDLE INCORRECT RESPONSE
335 LOCATE 24,2
          :PRINT WANSWER$;
340 LOCATE 24,1
345 WCOUNT%=WCOUNT%+1
350 WQUEST%(WCOUNT%)=QNUM%(QMAT%)
355 WQUEST$(WCOUNT%)=A$
360 RETURN
365 '
370 '   SUBROUTINE - HANDLE CORRECT RESPONSE
375 LOCATE 24,2
          :PRINT RANSWER$;
380 LOCATE 24,1
385 RCOUNT%=RCOUNT%+1
390 RETURN
395 '
10000 '**********************************************
10005 '*   CONSTANT TABLE AND DEFINED FUNCTIONS     *
10010 '*          (LINES 10000-10999)               *
10015 '**********************************************
10020 DASH$=STRING$(80,45)
10025 ENTER$=STRING$(22,32)+"Press <ENTER> to continue."
10030 PROMPT$=STRING$(22,32)+"Press <letter> of your choice
      ."
10035 YNINST$=STRING$(23,32)+"Press <Y>es or <N>o to continu
      e."
10040 QDATA$=STRING$(14,32)+"Enter requested data and press
      <ENTER> to continue."
10045 RANSWER$="Right response!"+STRING$(14,32)+"Press <ENTE
      R> to continue."
```

```
10050 WANSWER$="Wrong response!"+STRING$(14,32)+"Press <ENTE
      R> to continue."
10055 '
11000 '*********************************************
11005 '*              MAIN PROGRAM                 *
11010 '*           (LINES 11000-39999)             *
11015 '*********************************************
11020 '
11025 OPTION BASE 1
          :KEY OFF
          :ON ERROR GOTO 12195
11030 DIM PAGE%(200),START%(200),MORE%(200),TYPE$(200)
11035 DIM TEXT$(20),WQUEST%(200),WQUEST$(200),QNUM%(200)
11040 '
11045 '   PRINT SCREEN AND GET USER NAME
11050 QNAME$="MAIN MENU"
          :QINST$=QDATA$
          :QPAGE$=""
          :GOSUB 125
11055 PRINT"ENSURE THAT THE CAPS LOCK KEY IS DEPRESSED AND R
      EMAINS"
11060 PRINT"DEPRESSED FOR THE DURATION OF THIS LESSON!"
11065 PRINT
11070 PRINT"Input your first initial and last name in capita
      l letters with"
11075 INPUT"no space (example: RSMITH):  ",STUDENT$
          :IF  STUDENT$="" THEN 11045
11080 '
11085 '   CHECK FOR CAPITAL INPUT - REPEAT IF NECESSARY
11090 TEST%=ASC(LEFT$(STUDENT$,1))
11095 IF TEST%>=65 AND TEST%<=90 THEN 11115
11100 PRINT
          :PRINT"Erroneous input...ensure that the caps lock
            key is depressed!"
          :PRINT
11105 GOTO 11070
11110 '
11115 '   OFFER TO RUN INTRODUCTORY LESSON
11120 QNAME$="MAIN MENU"
          :QINST$=YNINST$
          :QPAGE$=""
          :GOSUB 125
11125 LOCATE 3,1
          :PRINT"Do you want to run a short introductory
            lesson about this"
11130 PRINT"computer-assisted instruction program before
       starting a lesson?"
11135 LOCATE 24,1
11140 GOSUB 210
11145 IF A$="N" THEN 11165
11150 TNAME$="INTRO"    '    RUN INTRO LESSON
11155 GOTO 11215
11160 '
```

```
11165 '   PRINT SCREEN, LESSON CATALOGUE, AND GET LESSON
           FILE NAME
11170 QNAME$="MAIN MENU"
           :QINST$=QDATA$
           :QPAGE$=""
           :QUEST$=""
           :GOSUB 125
11175 LOCATE 3,1:PRINT"The following lesson files are availa
      ble on this disk:"
11180 PRINT
11185 FILES "A:*.TXT"
           :LOCATE CSRLIN-1,1
           :PRINT STRING$(80-POS(0),32)
11190 PRINT
           :PRINT
           :INPUT"Enter lesson to run or change data disk and
            press <ENTER>: ",TNAME$
11195 IF TNAME$="" THEN 11165   'NEW DISK CATALOG
11200 IF TNAME$="Q" THEN 12145
11205 IF INSTR(TNAME$,".")=0 THEN 11215
11210 TNAME$=LEFT$(TNAME$,(INSTR(TNAME$,".")-1))
11215 TEXT$="A:"+TNAME$+".TXT"
11220 TABLE$="A:"+TNAME$+".TAB"
11225 '
11230 '   CHECK FOR LESSON TEXT FILE ON DISK, PRINT ERROR
           MESSAGE, AND
11235 '   RETURN TO MAIN MENU IF NECESSARY
11240 QTEST$=TEXT$
           :GOSUB 150
11245 IF FLAG%=0 THEN 11310
11250 QNAME$="ERROR"
           :QINST$=ENTER$
           :QPAGE$=""
           :GOSUB 125
11255 ON FLAG% GOTO 11260,11265,11275,11280
11260 PRINT TNAME$;" is not on these disks."
           :GOTO 11290
11265 PRINT"A device input/output error has occurred!"
11270 PRINT"Terminate this session and contact your training
      supervisor."
           :GOTO 11290
11275 PRINT TNAME$;" is not a valid lesson name."
           :GOTO 11290
11280 PRINT"An unknown disk error has occurred!"
11285 PRINT"Terminate this session and contact your training
       supervisor."
           :GOTO 11290
11290 LOCATE 24,1
11295 GOSUB 300
11300 GOTO 11165
11305 '
11310 '   CHECK FOR LESSON TABLE FILE ON DISK, PRINT ERROR
           MESSAGE, AND
```

```
11315 '   RETURN TO MAIN MENU IF REQUIRED
11320 QTEST$=TABLE$
        :GOSUB 150
11325 IF FLAG%=0 THEN 11380
11330 QNAME$="ERROR"
        :QINST$=ENTER$
        :QPAGE$=""
        :GOSUB 125
11335 PRINT"Unable to run this lesson...the corresponding le
      sson table"
11340 PRINT"file is not on these disks."
11345 PRINT
        :PRINT"Contact your training supervisor to make co
         rrections to"
11350 PRINT"this lesson disk."
11355 PRINT
        :PRINT"You may try another lesson or terminate thi
         s session."
11360 LOCATE 24,1
11365 GOSUB 300
11370 GOTO 11165
11375 '
11380 '   INITIALIZE QUESTION COUNTERS AND ARRAYS
11385 TCOUNT%=0
        :RCOUNT%=0
        :WCOUNT%=0
11390 ERASE PAGE%,START%,MORE%,TYPE$
11395 ERASE TEXT$,WQUEST%,WQUEST$,QNUM%
11400 DIM PAGE%(200),START%(200),MORE%(200),TYPE$(200)
11405 DIM TEXT$(20),WQUEST%(200),WQUEST$(200),QNUM%(200)
11410 '   READ LESSON TABLE
11415 QNAME$=TNAME$
        :QINST$=STRING$(20,32)+"Loading lesson table...ple
         ase wait..."
        :QPAGE$=""
        :GOSUB 125
11420 LOCATE 24,1
11425 OPEN TABLE$ FOR INPUT AS #1
11430 I=1
11435 IF EOF(1) THEN 11455
11440 INPUT #1,PAGE%(I),START%(I),MORE%(I),TYPE$(I)
11445 I=I+1
11450 GOTO 11435
11455 CLOSE
11460 NUMPAGES%=I-1
11465 '   SORT TABLE INTO PAGE # SEQUENCE
11470 FOR I=1 TO NUMPAGES%-1
11475 FOR J=I+1 TO NUMPAGES%
11480 IF PAGE%(I)>PAGE%(J) THEN 11485 ELSE 11505
11485 SWAP PAGE%(I),PAGE%(J)
11490 SWAP START%(I),START%(J)
11495 SWAP MORE%(I),MORE%(J)
11500 SWAP TYPE$(I),TYPE$(J)
```

```
11505 NEXT J
11510 NEXT I
11515 QNUM%=1
11520 FOR I=2 TO NUMPAGES%
11525 IF TYPE$(I)<>"?" THEN 11540
11530 QNUM%(I)=QNUM%
11535 QNUM%=QNUM%+1
11540 NEXT I
11545 '
11550 '   OPEN LESSON BUFFER AND START LESSON WITH PAGE 1
11555 OPEN TEXT$ AS #2 LEN=81
11560 FIELD #2,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5
       AS BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,
      46 AS DUMMY$
11565 FIELD #2,1 AS BUF10$,80 AS BUF11$
11570 QPAGE%=1   'SET COUNTER FOR FIRST PAGE OF LESSON
11575 '
11580 '   FIND CORRESPONDING TABLE SUBSCRIPT
11585 COUNT%=1
11590 IF PAGE%(COUNT%)=QPAGE% THEN 11605
11595 COUNT%=COUNT%+1
11600 GOTO 11590
11605 QMAT%=COUNT%
11610 '
11615 '   READ DATA FOR A PAGE
11620 GET #2,START%(QMAT%)
11625 TYPE$=BUF1$
11630 AJUMP%=VAL(BUF3$)
11635 BJUMP%=VAL(BUF4$)
11640 CJUMP%=VAL(BUF5$)
11645 DJUMP%=VAL(BUF6$)
11650 EJUMP%=VAL(BUF7$)
11655 ANSWER$=BUF8$
11660 MORE%=VAL(BUF9$)
11665 '
11670 '   READ THE TEXT RECORDS FOR THIS PAGE
11675 FOR I=1 TO MORE%
11680 GET #2,(START%(QMAT%)+I)
11685 DUMMY$=BUF10$
11690 TEXT$(I)=BUF11$
11695 NEXT I
11700 '
11705 '   DECIDE ON PROPER INSTRUCTION LINE AND DISPLAY PAGE
11710 IF TYPE$="#" THEN QINST$=ENTER$
11715 IF TYPE$="?" THEN QINST$=PROMPT$
11720 QPAGE$=STR$(QPAGE%)
11725 QNAME$=TNAME$
11730 IF TYPE$="?" THEN TCOUNT%=TCOUNT%+1
11735 IF TYPE$="?" THEN QUEST$="QUESTION #"+STR$(QNUM%(QMAT%
      )) ELSE QUEST$=""
11740 GOSUB 125
11745 FOR I=1 TO MORE%
11750 PRINT TEXT$(I);
```

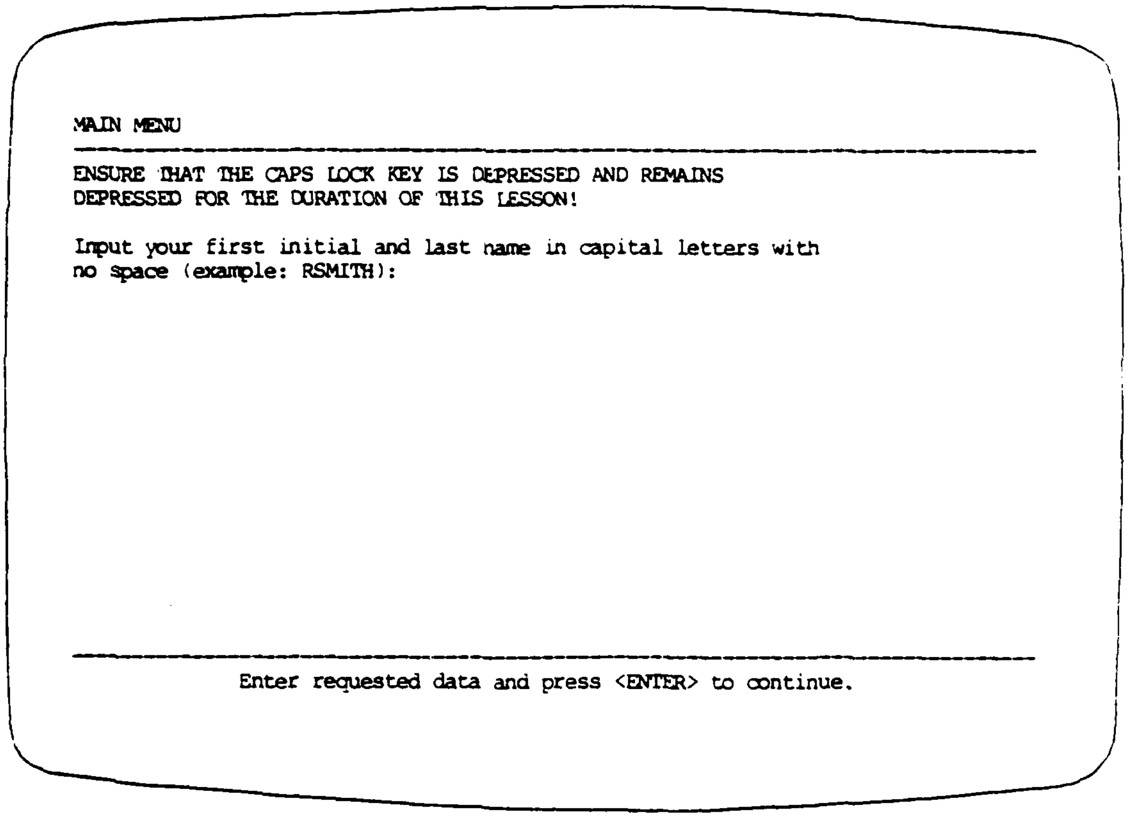
```
11755 NEXT I
11760 LOCATE 24,1
11765 '
11770 '   ROUTE TO PROPER LINE FOR HANDLING RESPONSE TO PAGE
          TYPE
11775 IF TYPE$="#" THEN 11795   'TEXT PAGE
11780 IF INSTR("ABCDE",ANSWER$)<>0 THEN 11820   'MULTIPLE
      CHOICE QUESTION
11785 IF INSTR("TF",ANSWER$)<>0 THEN 11880   'TRUE/FALSE
      QUESTION
11790 '
11795 '   HANDLE TEXT PAGE USER RESPONSE
11800 GOSUB 300
11805 QPAGE%=AJUMP%
11810 GOTO 11925
11815 '
11820 '   HANDLE MULTIPLE CHOICE QUESTION RESPONSE
11825 GOSUB 240
11830 IF A$="A" THEN QPAGE%=AJUMP%
11835 IF A$="B" THEN QPAGE%=BJUMP%
11840 IF A$="C" THEN QPAGE%=CJUMP%
11845 IF A$="D" THEN QPAGE%=DJUMP%
11850 IF A$="E" THEN QPAGE%=EJUMP%
11855 IF A$<>ANSWER$ THEN GOSUB 330
11860 IF A$=ANSWER$ THEN GOSUB 370
11865 GOSUB 300
11870 GOTO 11925
11875 '
11880 '   HANDLE TRUE/FALSE QUESTION RESPONSE
11885 GOSUB 270
11890 IF A$="T" THEN QPAGE%=AJUMP%
11895 IF A$="F" THEN QPAGE%=BJUMP%
11900 IF A$<>ANSWER$ THEN GOSUB 330
11905 IF A$=ANSWER$ THEN GOSUB 370
11910 GOSUB 300
11915 GOTO 11925
11920 '
11925 '   CHECK FOR BRANCH TO LAST PAGE
11930 IF QPAGE%=9999 THEN 11945
11935 GOTO 11580   '   REPEAT PROCESS FOR NEW PAGE
11940 '
11945 '   PRINT SCREEN WITH STUDENT RESULTS
11950 QNAME$=TNAME$
          :QINST$=ENTER$
          :QUEST$=""
          :QPAGE$=""
          :GOSUB 125
11955 GRADE%=(RCOUNT%/TCOUNT%)*100
11960 PRINT"During this lesson you were asked ";
11965 PRINT USING "###";TCOUNT%;
11970 PRINT" questions."
11975 PRINT
11980 PRINT"You answered ";
```

```
11985 PRINT USING "###";RCOUNT%;
11990 PRINT" correctly and ";
11995 PRINT USING "###";WCOUNT%;
12000 PRINT" incorrectly."
12005 PRINT
12010 PRINT"Your grade for this lesson is ";
12015 PRINT USING "###.";GRADE%
        :LOCATE 24,1
12020 GOSUB 300
12025 '
12030 '   END LESSON ROUTINE - WRITE STUDENT FILE
12035 CLOSE
        :IF TNAME$="INTRO" THEN 11165
12040 QNAME$="ENDING LESSON"
        :QPAGE$=""
        :QINST$=STRING$(20,32)+"Saving lesson results...pl
         ease wait."
        :QUEST$=""
        :GOSUB 125
12045 LOCATE 24,1
12050 '
12055 '   CHECK FOR EXISTING STUDENT FILE ON DRIVE 1 - OPEN
          IN CORRECT MODE
12060 '   WRITE DATA TO FILE
12065 STUFILE$="A:"+TNAME$+".STU"
12070 QTEST$=STUFILE$
12075 GOSUB 150
12080 IF FLAG%=0 THEN OPEN STUFILE$ FOR APPEND AS #1
12085 IF FLAG%=1 THEN OPEN STUFILE$ FOR OUTPUT AS #1
12090 WRITE# 1, STUDENT$,DATE$,TCOUNT%,RCOUNT%,WCOUNT%
12095 FOR I=1 TO WCOUNT%
12100 WRITE# 1,WQUEST%(I),WQUEST$(I)
12105 NEXT I
12110 CLOSE
12115 '
12120 '   RETURN TO MAIN MENU
12125 GOTO 11165
12130 '
12135 '   TERMINATE SESSION ROUTINE
12140 CLOSE
12145 QNAME$="GOODBYE"
        :QINST$=""
        :QPAGE$=""
        :QUEST$=""
        :GOSUB 125
12150 PRINT "Thank you for using the LEARNER computer-assist
      ed instruction"
12155 PRINT"system."
12160 PRINT
12165 PRINT"Wait for the DOS ready prompt before removing di
      sks or turning"
12170 PRINT"machine off!"
12175 LOCATE 24,1
```

```
12180 FOR I=1 TO 3000:NEXT I    '    DELAY LOOP TO DISPLAY
      LOGOFF MESSAGE
12185 CLOSE
          :SYSTEM
          :END
12190 '
12195 '   PROGRAM FATAL ERROR ROUTINE
12200 STAR$=STRING$(10,32)+STRING$(60,42)
12205 CLS
12210 LOCATE 4,1
          :PRINT STAR$
          :FOR I=5 TO 16
          :LOCATE I,11
          :PRINT"**";
          :LOCATE I,69
          :PRINT"**";
          :NEXT I
          :LOCATE 16,1
          :PRINT STAR$;
12215 LOCATE 6,22
          :PRINT"FATAL PROGRAM ERROR DURING EXECUTION";
12220 LOCATE 8,26
          :PRINT"Error code ";ERR;" in line ";ERL;
12225 LOCATE 11,16
          :PRINT"Copy above data and deliver to training sup
           ervisor!";
12230 LOCATE 14,24
          :PRINT"Press <ENTER> to restart program.";
12235 LOCATE 14,23
12240 A$=INKEY$
          :IF A$<>CHR$(13) THEN 12240
12245 CLOSE
          :RESUME 11045
12250 END
```

Appendix B: LEARNER/BAS Courseware Example

This appendix contains example courseware generated by the LEARNER/BAS program. Figure B.1 contains hardcopy prints of screens as they appear during lesson execution. The courseware illustrated is the introductory lesson. Although all possible screens have not been shown, a representative sample has been illustrated.



MAIN MENU

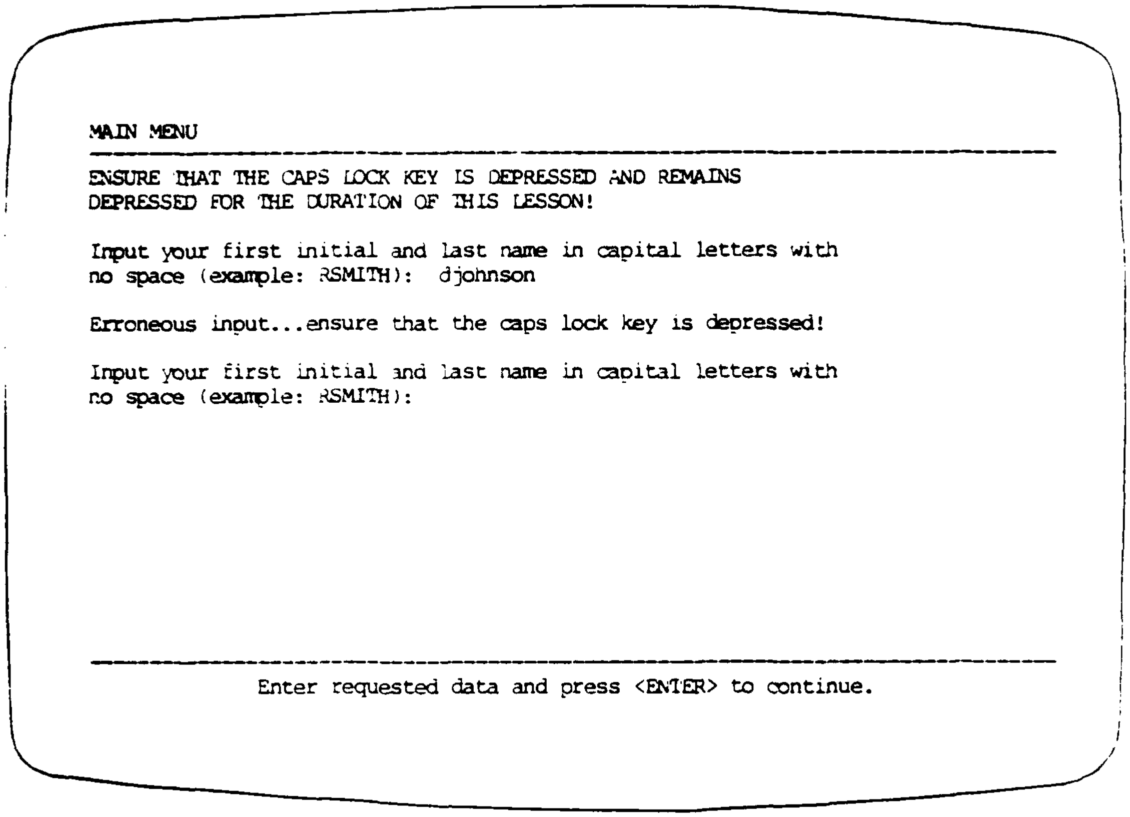
ENSURE THAT THE CAPS LOCK KEY IS DEPRESSED AND REMAINS
DEPRESSED FOR THE DURATION OF THIS LESSON!

Input your first initial and last name in capital letters with
no space (example: RSMITH):

Enter requested data and press <ENTER> to continue.

LEARNER/BAS main menu and input of student name. Throughout lesson, lesson information appears on top line of screen and user instructions on bottom line of screen.

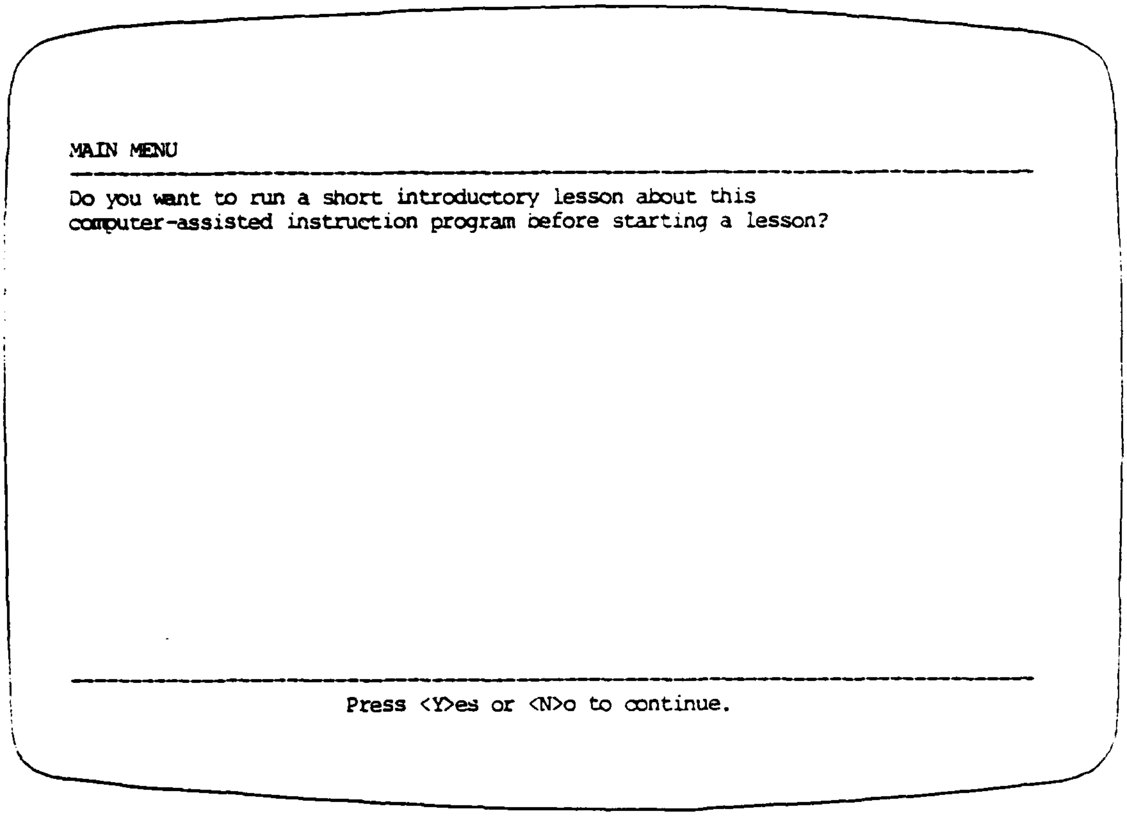
Figure B.1: LEARNER/BAS Courseware Example



```
MAIN MENU
--------------------------------------------------------------------------------
ENSURE THAT THE CAPS LOCK KEY IS DEPRESSED AND REMAINS
DEPRESSED FOR THE DURATION OF THIS LESSON!

Input your first initial and last name in capital letters with
no space (example: RSMITH):  djohnson

Erroneous input...ensure that the caps lock key is depressed!

Input your first initial and last name in capital letters with
no space (example: RSMITH):









--------------------------------------------------------------------------------
              Enter requested data and press <ENTER> to continue.
```

Program checks initial character of student name for upper-case input. If not upper-case this error message is displayed and request for student name displayed again.

Figure B.1: LEARNER/BAS Courseware Example (continued)



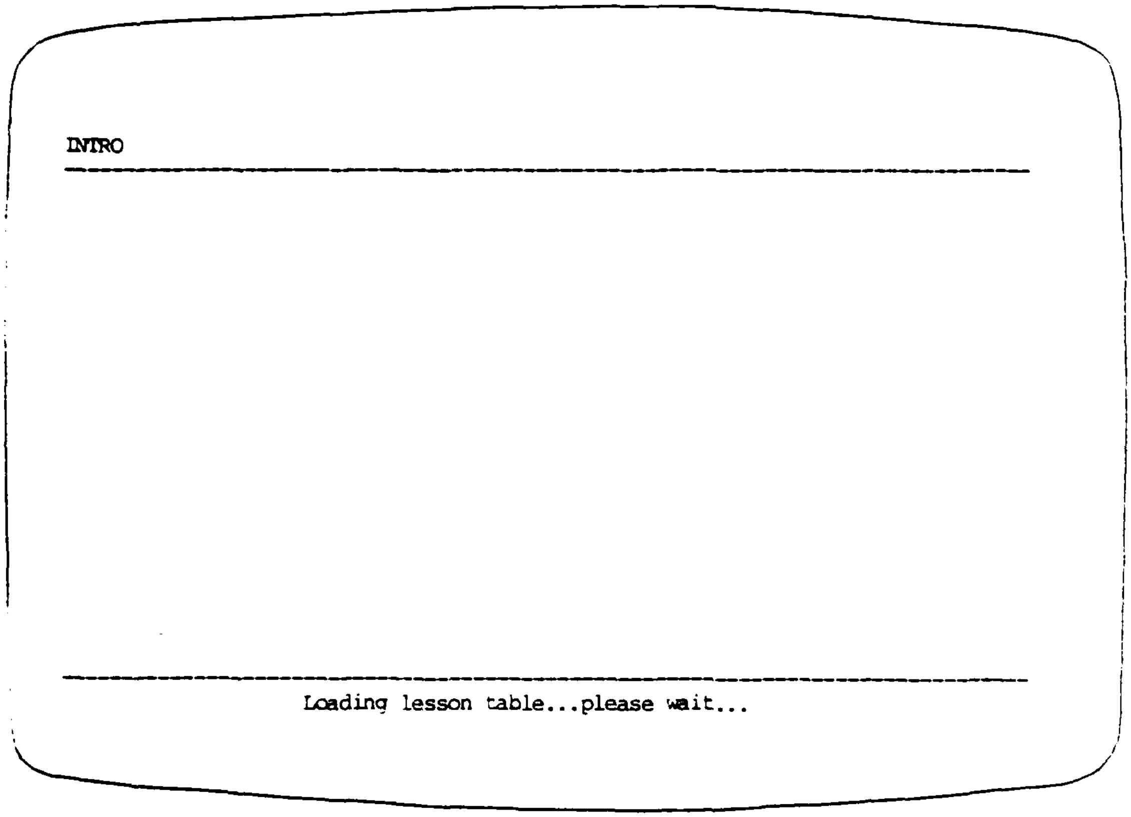
MAIN MENU

Do you want to run a short introductory lesson about this computer-assisted instruction program before starting a lesson?

Press <Y>es or <N>o to continue.

Program asks if introductory lesson is to be run. If student answers with <N> key then program branches to display of lesson files available on disk.

Figure B.1: LEARNER/BAS Courseware Example (continued)

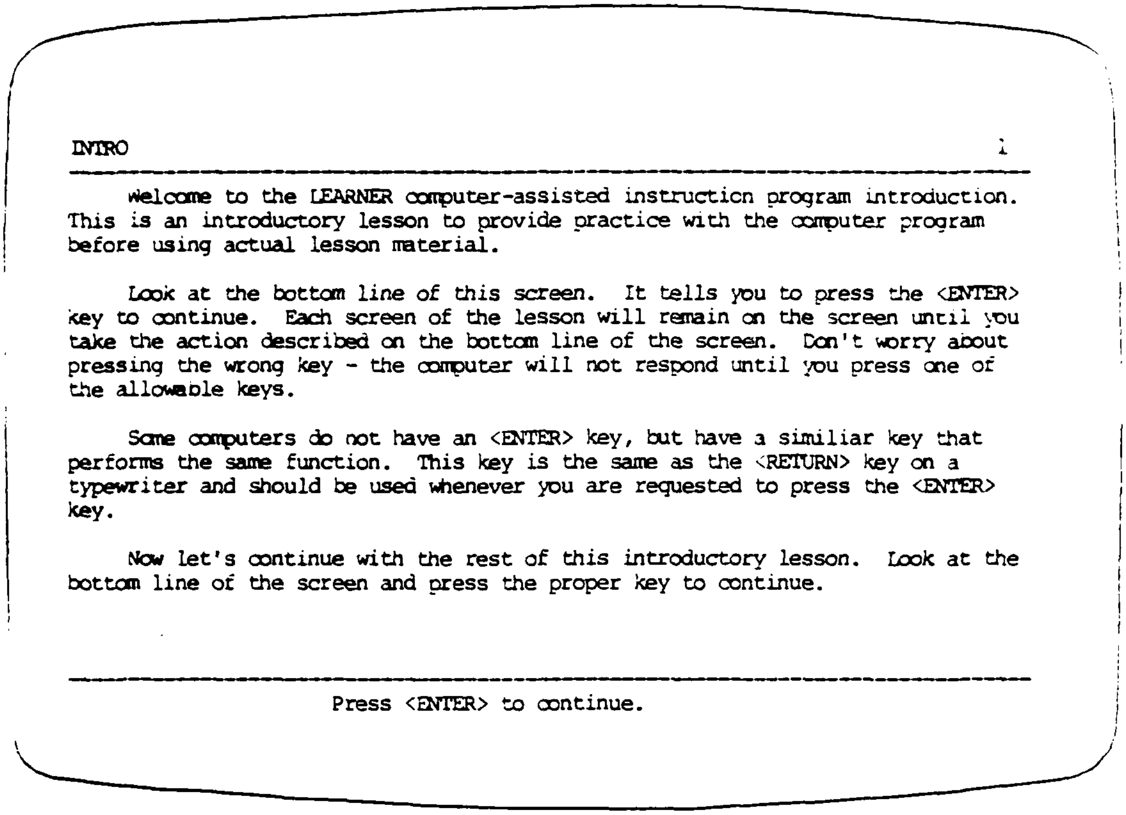


INTRO

Loading lesson table...please wait...

Temporary screen display while lesson table file is being loaded into computer memory disk.

Figure B.1: LEARNER/BAS Courseware Example (continued)



INTRO 1

Welcome to the LEARNER computer-assisted instruction program introduction. This is an introductory lesson to provide practice with the computer program before using actual lesson material.

Look at the bottom line of this screen. It tells you to press the <ENTER> key to continue. Each screen of the lesson will remain on the screen until you take the action described on the bottom line of the screen. Don't worry about pressing the wrong key - the computer will not respond until you press one of the allowable keys.

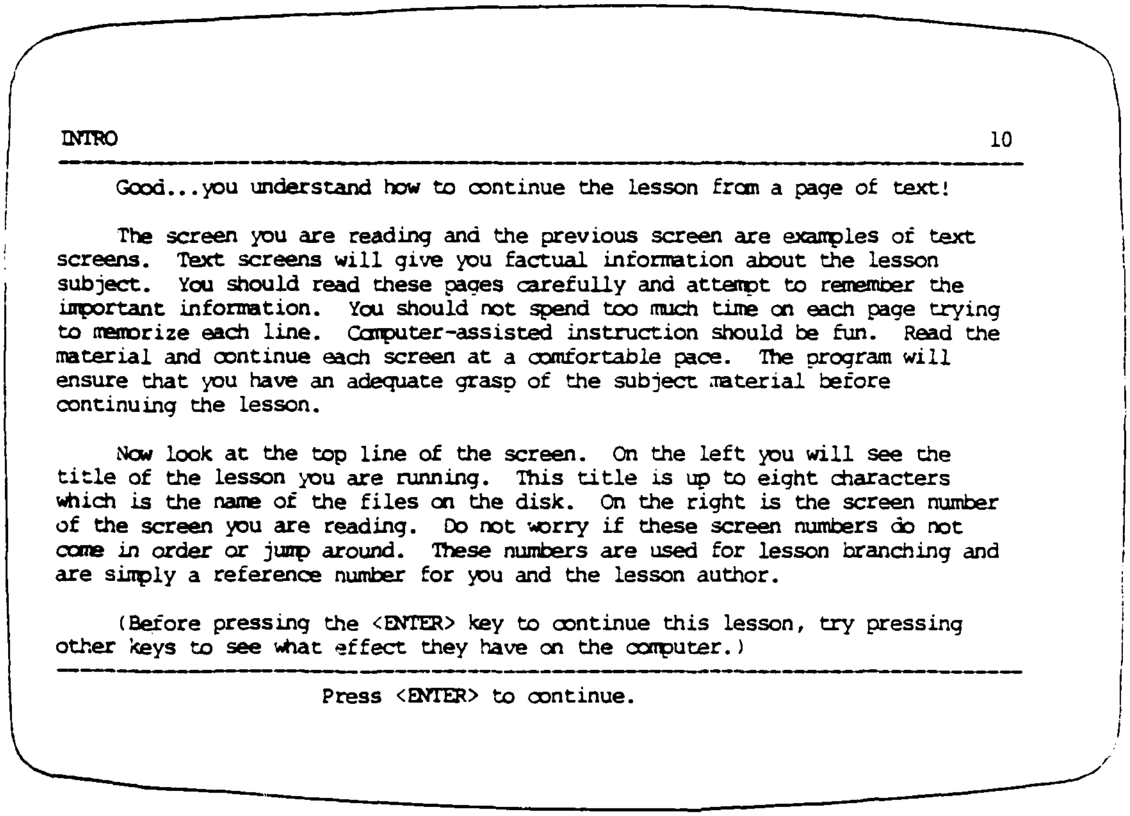
Some computers do not have an <ENTER> key, but have a similiar key that performs the same function. This key is the same as the <RETURN> key on a typewriter and should be used whenever you are requested to press the <ENTER> key.

Now let's continue with the rest of this introductory lesson. Look at the bottom line of the screen and press the proper key to continue.

Press <ENTER> to continue.

First screen of the introductory lesson illustrating a typical text page. Note that the lesson information line displays the lesson filename being executed and the lesson screen page number.

Figure B.1: LEARNER/BAS Courseware Example (continued)



INTRO 10

Good...you understand how to continue the lesson from a page of text!

The screen you are reading and the previous screen are examples of text screens. Text screens will give you factual information about the lesson subject. You should read these pages carefully and attempt to remember the important information. You should not spend too much time on each page trying to memorize each line. Computer-assisted instruction should be fun. Read the material and continue each screen at a comfortable pace. The program will ensure that you have an adequate grasp of the subject material before continuing the lesson.

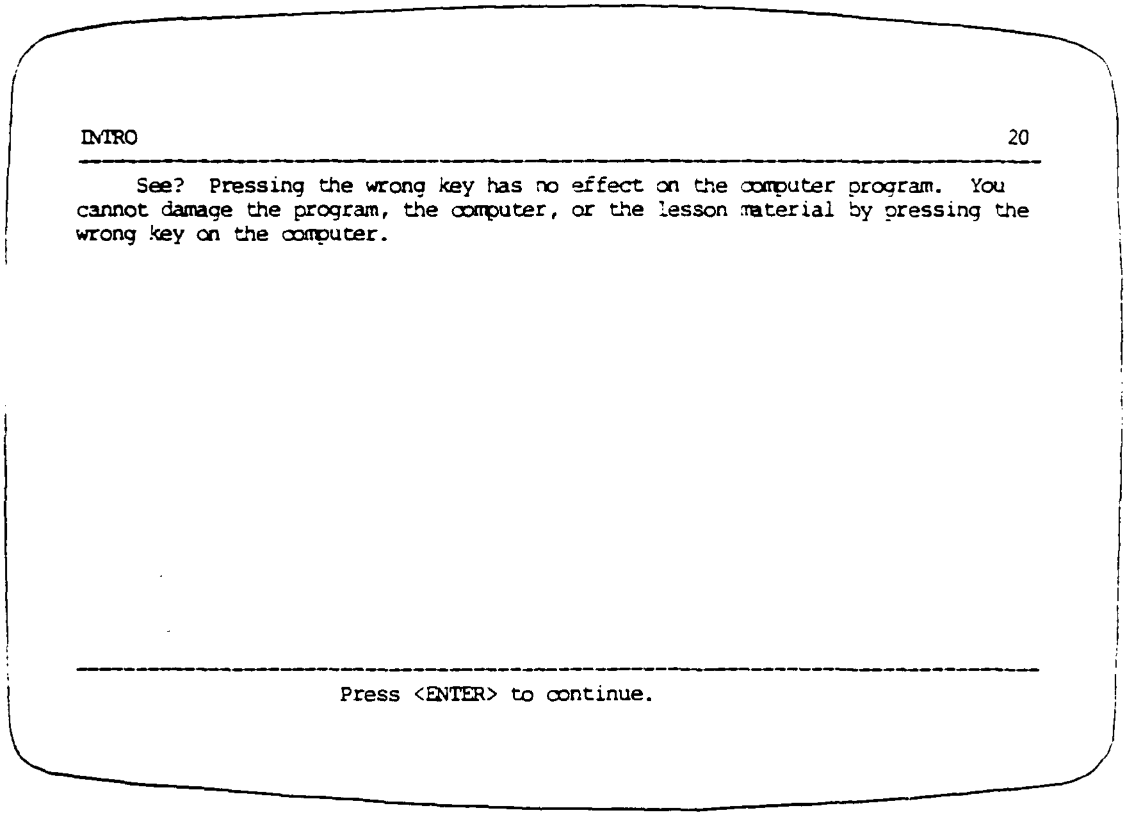
Now look at the top line of the screen. On the left you will see the title of the lesson you are running. This title is up to eight characters which is the name of the files on the disk. On the right is the screen number of the screen you are reading. Do not worry if these screen numbers do not come in order or jump around. These numbers are used for lesson branching and are simply a reference number for you and the lesson author.

(Before pressing the <ENTER> key to continue this lesson, try pressing other keys to see what effect they have on the computer.)

Press <ENTER> to continue.

Second screen of the introductory lesson.

Figure B.1: LEARNER/BAS Courseware Example (continued)



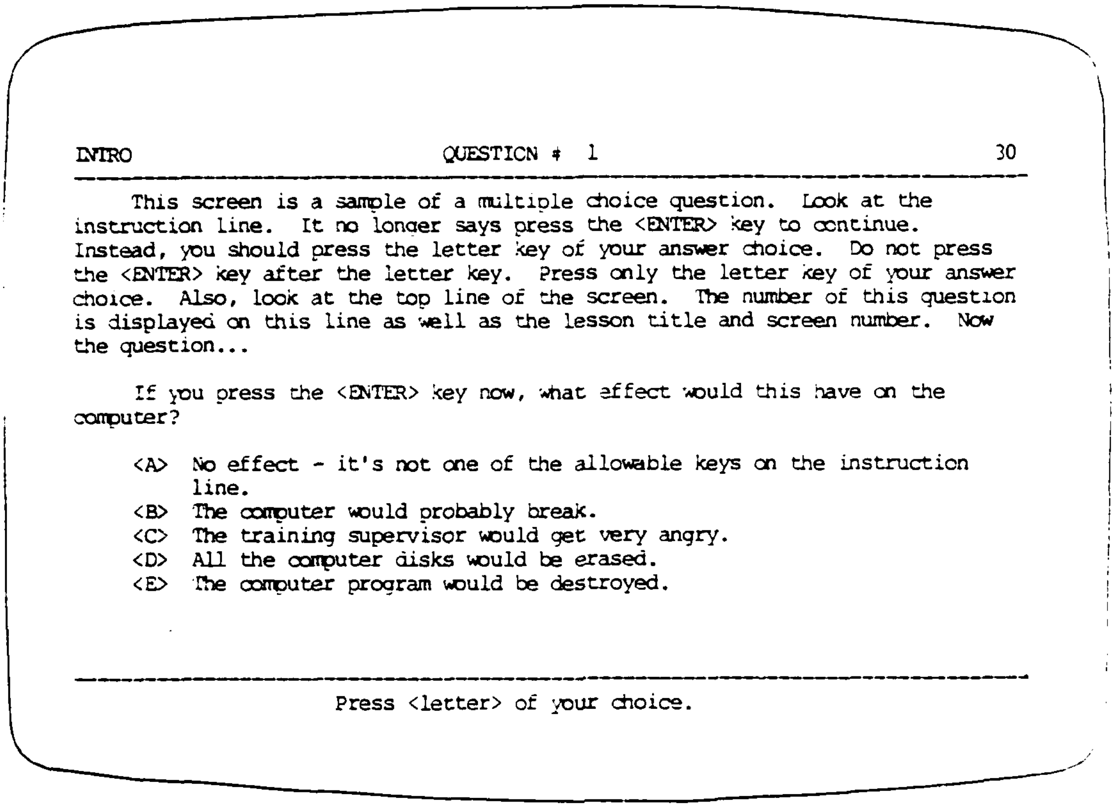
INTRO 20

See? Pressing the wrong key has no effect on the computer program. You cannot damage the program, the computer, or the lesson material by pressing the wrong key on the computer.

Press <ENTER> to continue.

Third screen of the introductory lesson.

Figure B.1: LEARNER/BAS Courseware Example (continued)

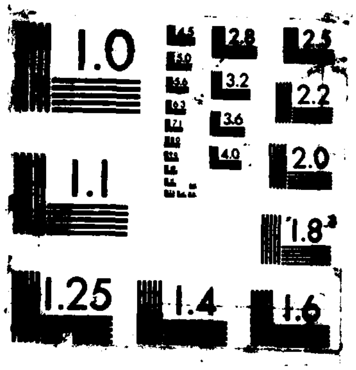


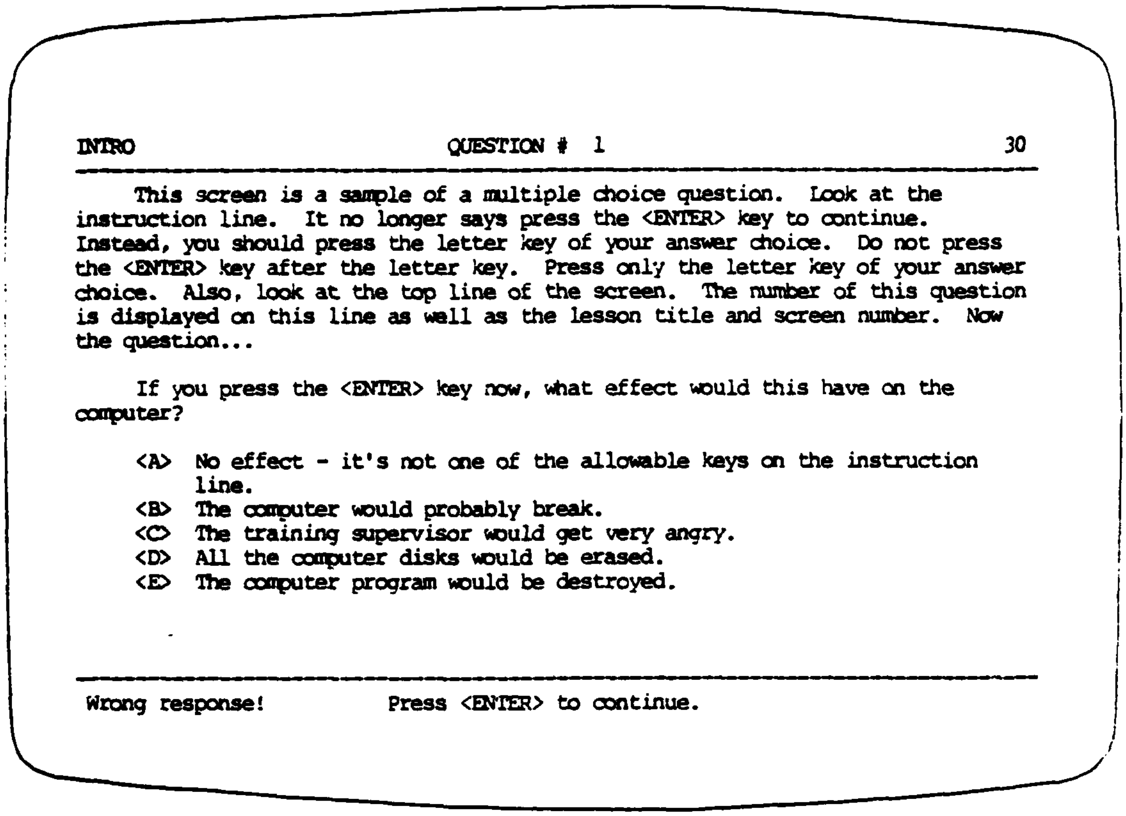
```
INTRO                         QUESTICN # 1                              30
--------------------------------------------------------------------------
     This screen is a sample of a multiple choice question.  Look at the
instruction line.  It no longer says press the <ENTER> key to continue.
Instead, you should press the letter key of your answer choice.  Do not press
the <ENTER> key after the letter key.  Press only the letter key of your answer
choice.  Also, look at the top line of the screen.  The number of this question
is displayed on this line as well as the lesson title and screen number.  Now
the question...

     If you press the <ENTER> key now, what effect would this have on the
computer?

     <A>  No effect - it's not one of the allowable keys on the instruction
          line.
     <B>  The computer would probably break.
     <C>  The training supervisor would get very angry.
     <D>  All the computer disks would be erased.
     <E>  The computer program would be destroyed.

--------------------------------------------------------------------------
                    Press <letter> of your choice.
```

Fourth screen of the introductory lesson illustrating a typical multiple-choice question page. This particular screen combines text information with [illegible] choice question. Note that the [illegible] is displayed on the information [illegible] instruction line requests [illegible] letter key.

Figure [illegible]

AD-A187 268 A GENERAL APPLICATION COMPUTER-ASSISTED INSTRUCTION 2/3
SYSTEM FOR MICROCOMPUTERS(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF SYST.. R MASON
UNCLASSIFIED SEP 87 AFIT/GLM/LSR/87S-45 F/G 12/5 NL

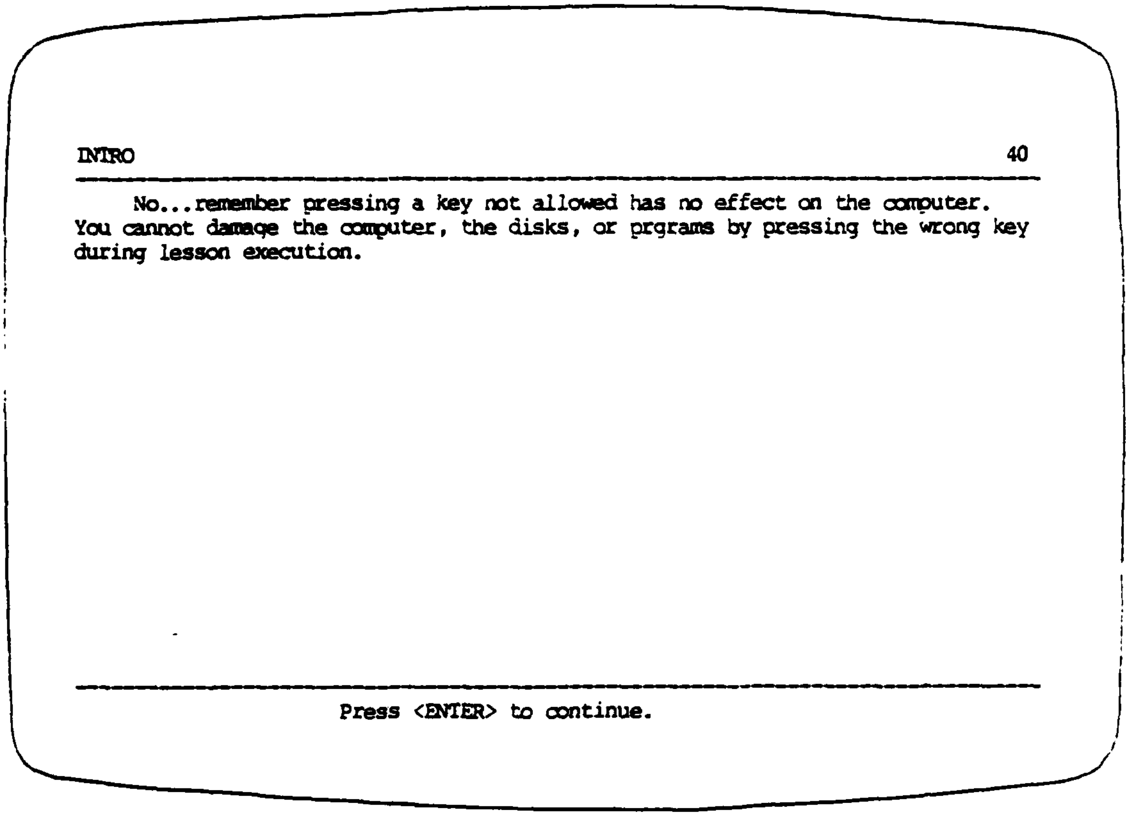

1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6



```
INTRO                          QUESTION # 1                              30
____________________________________________________________________________
     This screen is a sample of a multiple choice question.  Look at the
instruction line.  It no longer says press the <ENTER> key to continue.
Instead, you should press the letter key of your answer choice.  Do not press
the <ENTER> key after the letter key.  Press only the letter key of your answer
choice.  Also, look at the top line of the screen.  The number of this question
is displayed on this line as well as the lesson title and screen number.  Now
the question...

     If you press the <ENTER> key now, what effect would this have on the
computer?

     <A>  No effect - it's not one of the allowable keys on the instruction
          line.
     <B>  The computer would probably break.
     <C>  The training supervisor would get very angry.
     <D>  All the computer disks would be erased.
     <E>  The computer program would be destroyed.

____________________________________________________________________________
 Wrong response!          Press <ENTER> to continue.
```

Fourth screen of the introductory lesson illustrating a typical multiple-choice question page following an incorrect question response by the student.

Figure B.1: LEARNER/BAS Courseware Example (continued)



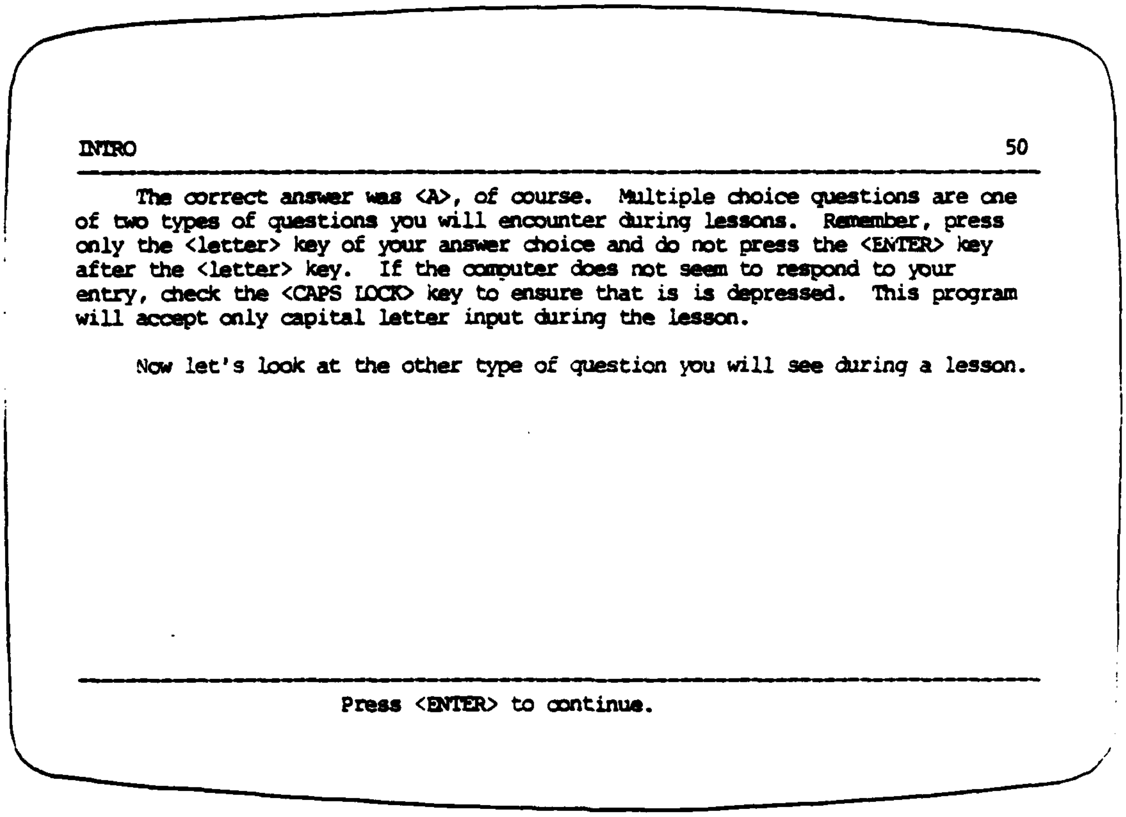
INTRO 40

No...remember pressing a key not allowed has no effect on the computer. You cannot damage the computer, the disks, or prgrams by pressing the wrong key during lesson execution.

Press <ENTER> to continue.

Screen displayed in the introductory lesson if student responded incorrectly to question #1. This screen illustrates the use of remedial information following an incorrect question response.

Figure B.1: LEARNER/BAS Courseware Example (continued)



INTRO 50

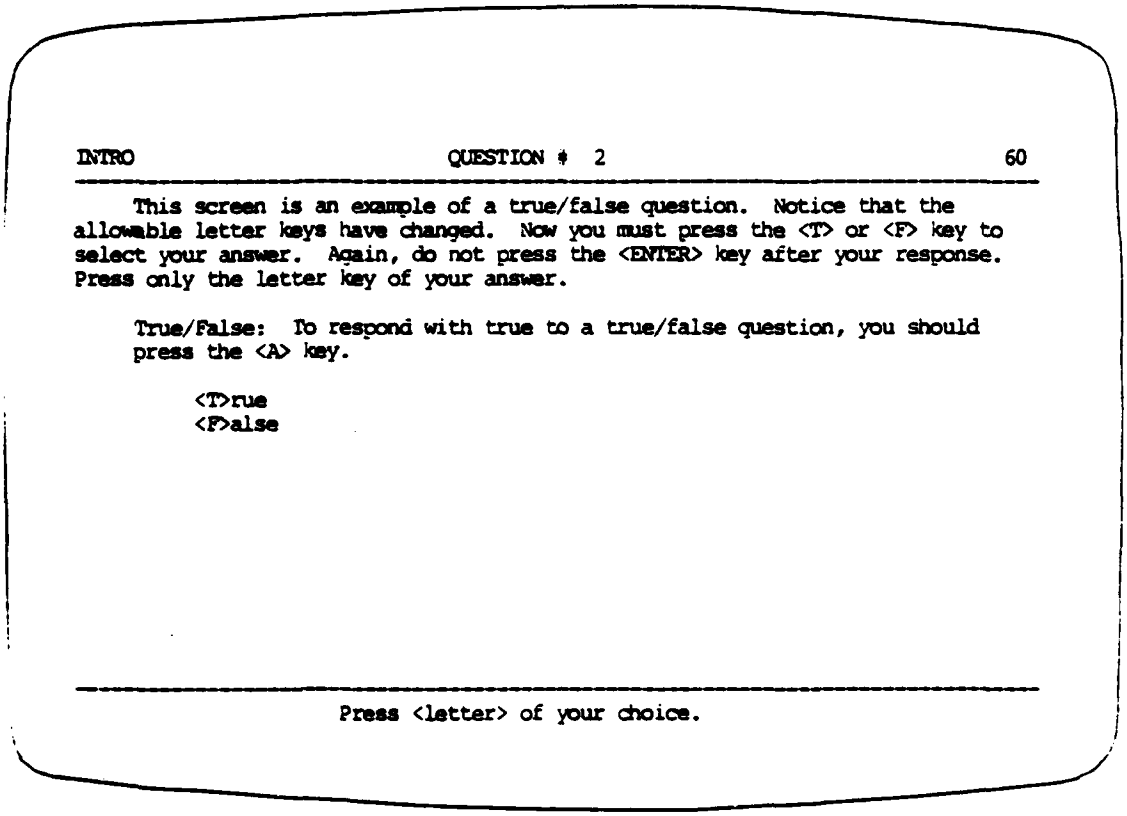
The correct answer was <A>, of course. Multiple choice questions are one of two types of questions you will encounter during lessons. Remember, press only the <letter> key of your answer choice and do not press the <ENTER> key after the <letter> key. If the computer does not seem to respond to your entry, check the <CAPS LOCK> key to ensure that is is depressed. This program will accept only capital letter input during the lesson.

Now let's look at the other type of question you will see during a lesson.

Press <ENTER> to continue.

Fifth screen of the introductory lesson.

Figure B.1: LEARNER/BAS Courseware Example (continued)



```
INTRO                         QUESTION # 2                        60
```

This screen is an example of a true/false question. Notice that the allowable letter keys have changed. Now you must press the <T> or <F> key to select your answer. Again, do not press the <ENTER> key after your response. Press only the letter key of your answer.

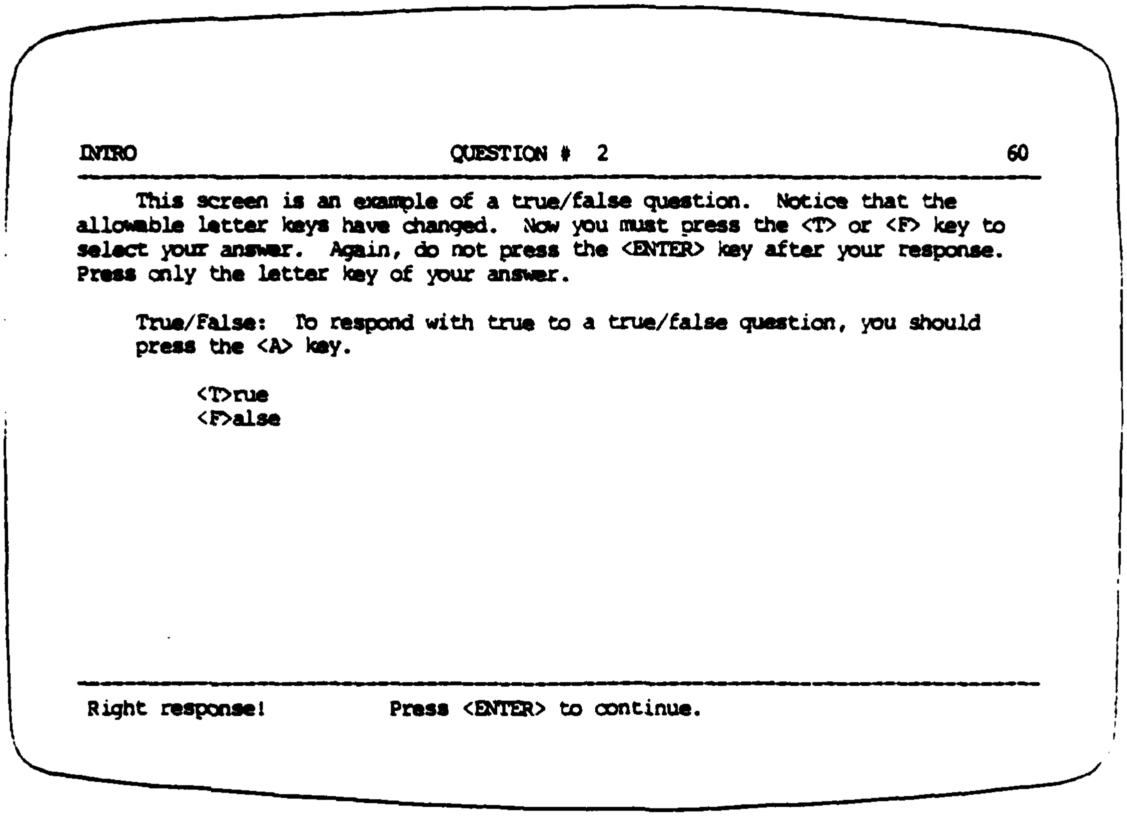
True/False: To respond with true to a true/false question, you should press the <A> key.

<T>rue
<F>alse

Press <letter> of your choice.

Sixth screen of the introductory lesson illustrating a typical true/false question screen. Again, this screen combines text information with the question.

Figure B.1: LEARNER/BAS Courseware Example (continued)



INTRO QUESTION # 2 60

This screen is an example of a true/false question. Notice that the allowable letter keys have changed. Now you must press the <T> or <F> key to select your answer. Again, do not press the <ENTER> key after your response. Press only the letter key of your answer.

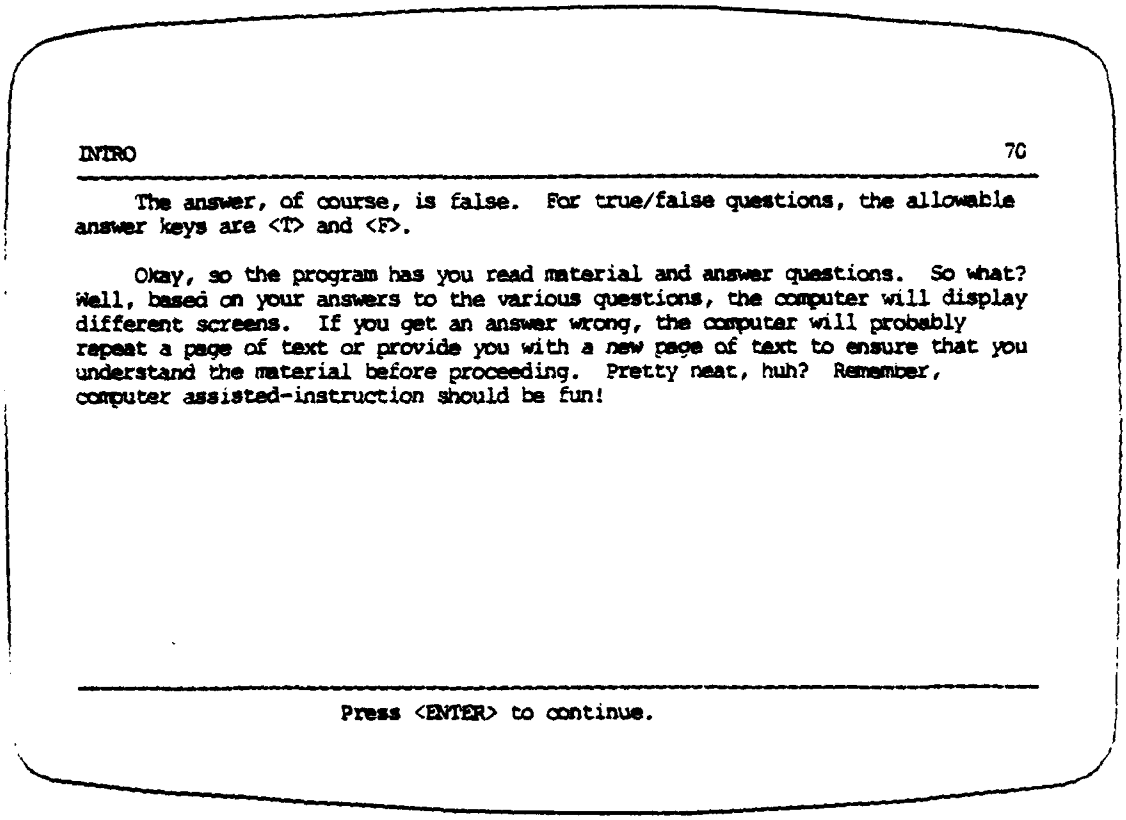
True/False: To respond with true to a true/false question, you should press the <A> key.

<T>rue
<F>alse

Right response! Press <ENTER> to continue.

Sixth screen of the introductory lesson illustrating a typical true/false question screen following a correct question response by the student.

Figure B.1: LEARNER/BAS Courseware Example (continued)



INTRO 7C

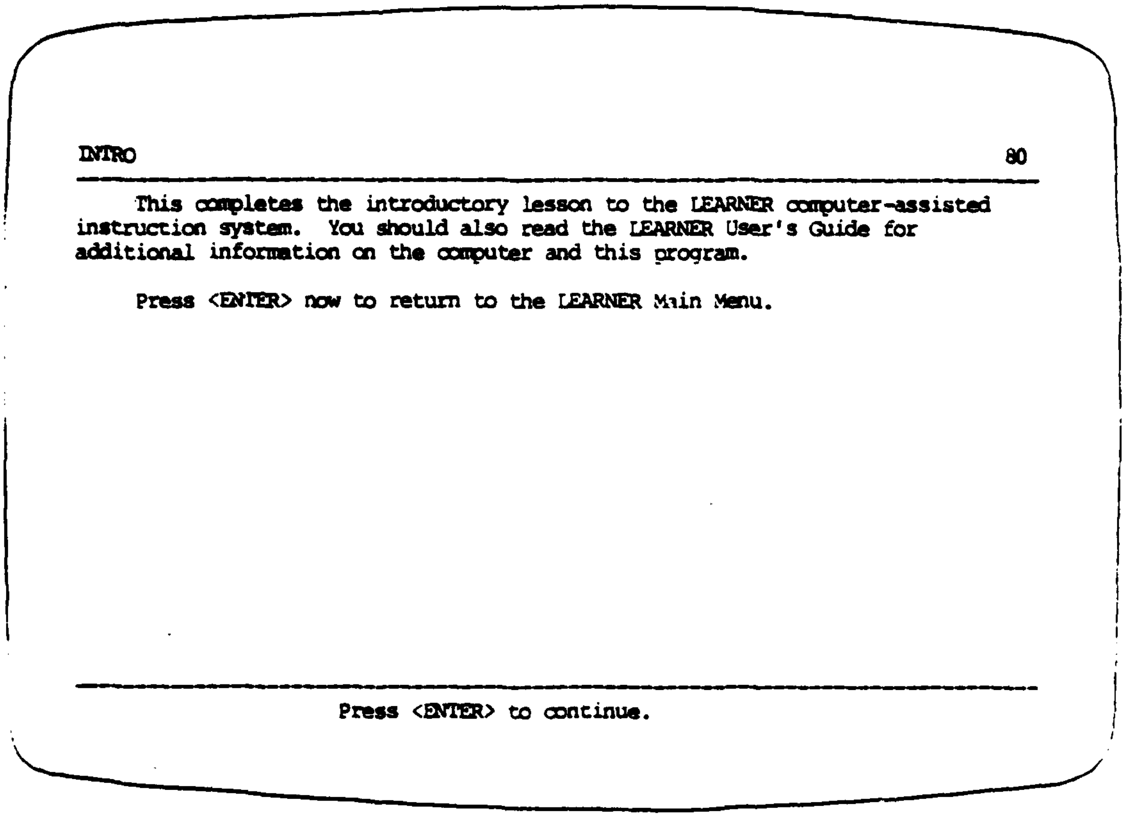
The answer, of course, is false. For true/false questions, the allowable answer keys are <T> and <F>.

Okay, so the program has you read material and answer questions. So what? Well, based on your answers to the various questions, the computer will display different screens. If you get an answer wrong, the computer will probably repeat a page of text or provide you with a new page of text to ensure that you understand the material before proceeding. Pretty neat, huh? Remember, computer assisted-instruction should be fun!

Press <ENTER> to continue.

Seventh screen of the introductory lesson.

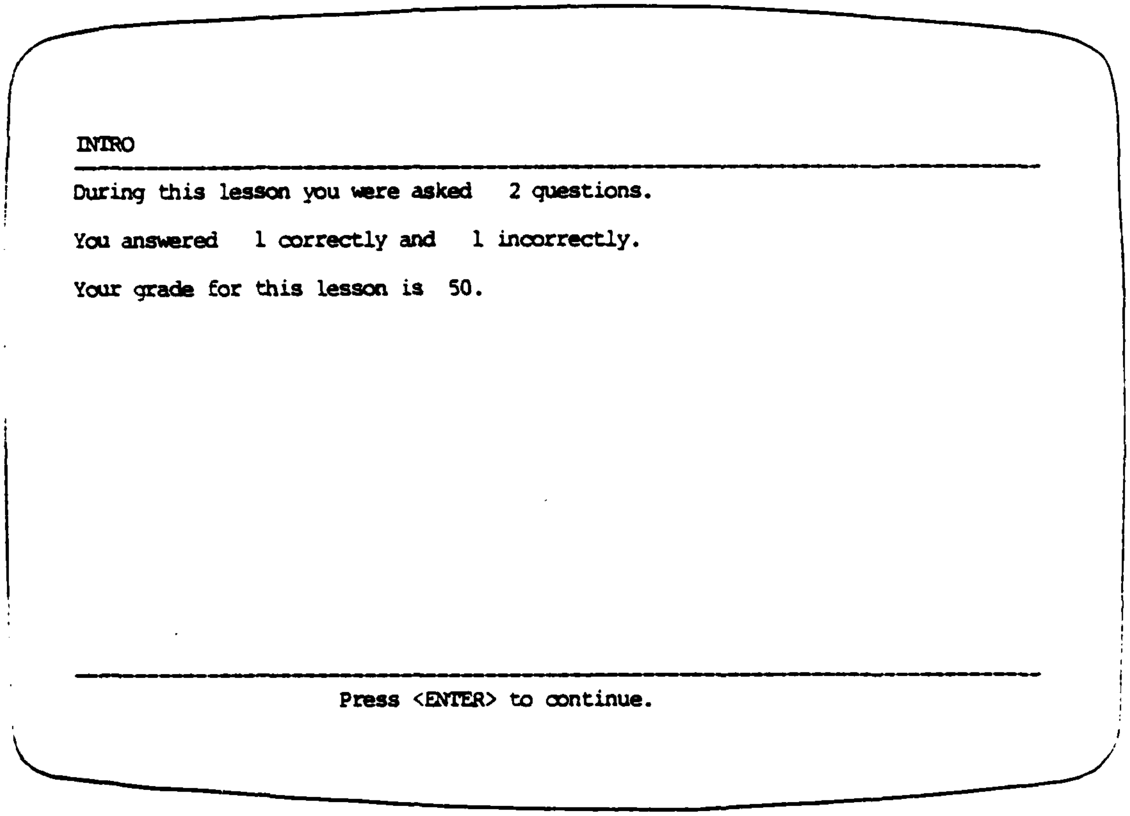
Figure B.1: LEARNER/BAS Courseware Example (continued)



```
INTRO                                                              80
_____________________________________________________________________
     This completes the introductory lesson to the LEARNER computer-assisted
instruction system.  You should also read the LEARNER User's Guide for
additional information on the computer and this program.

     Press <ENTER> now to return to the LEARNER Main Menu.









_____________________________________________________________________
                        Press <ENTER> to continue.
```

Final screen of the introductory lesson. The jump page for this screen is "9999" which terminates the lesson execution and branches to the end of lesson routine.

Figure B.1: LEARNER/BAS Courseware Example (continued)



INTRO

During this lesson you were asked 2 questions.

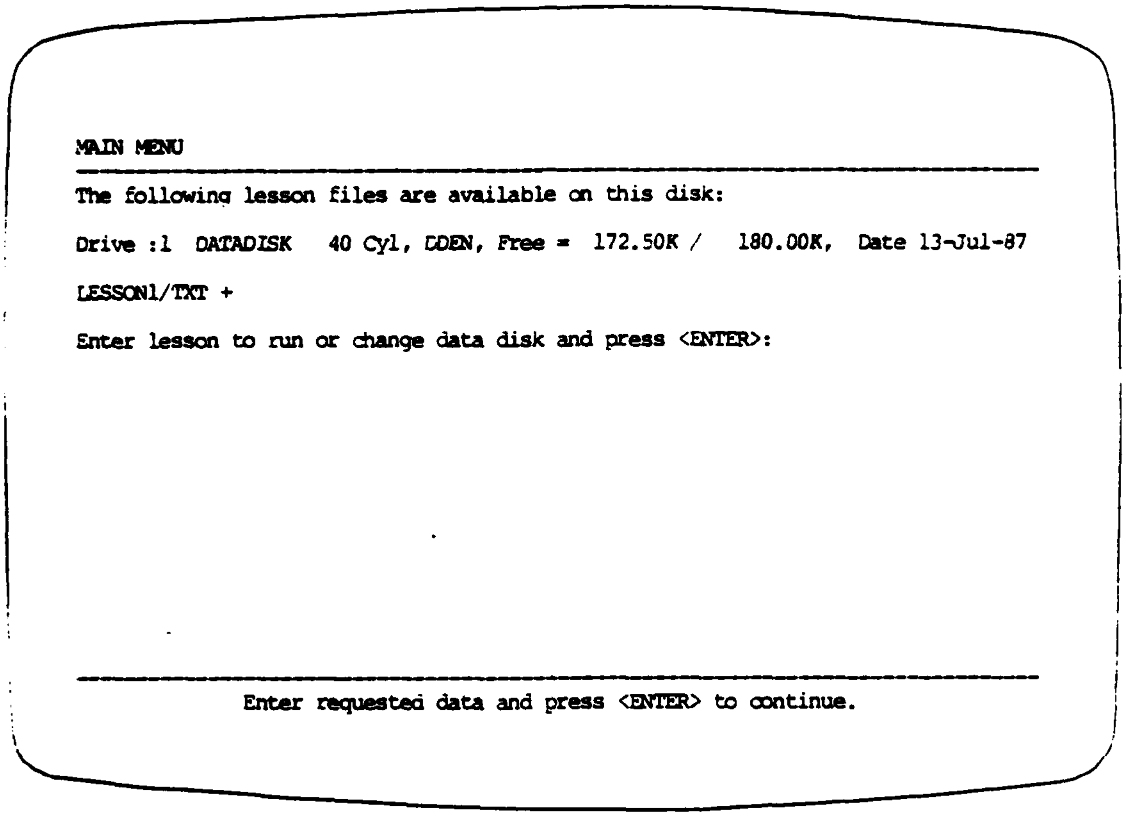
You answered 1 correctly and 1 incorrectly.

Your grade for this lesson is 50.

Press <ENTER> to continue.

Display of student information provided after lesson termination. The total questions asked will probably not match the total questions in the lesson due to repeated question screens in the lesson branching instructions. This information is also recorded in the student disk file for analysis by the courseware author.

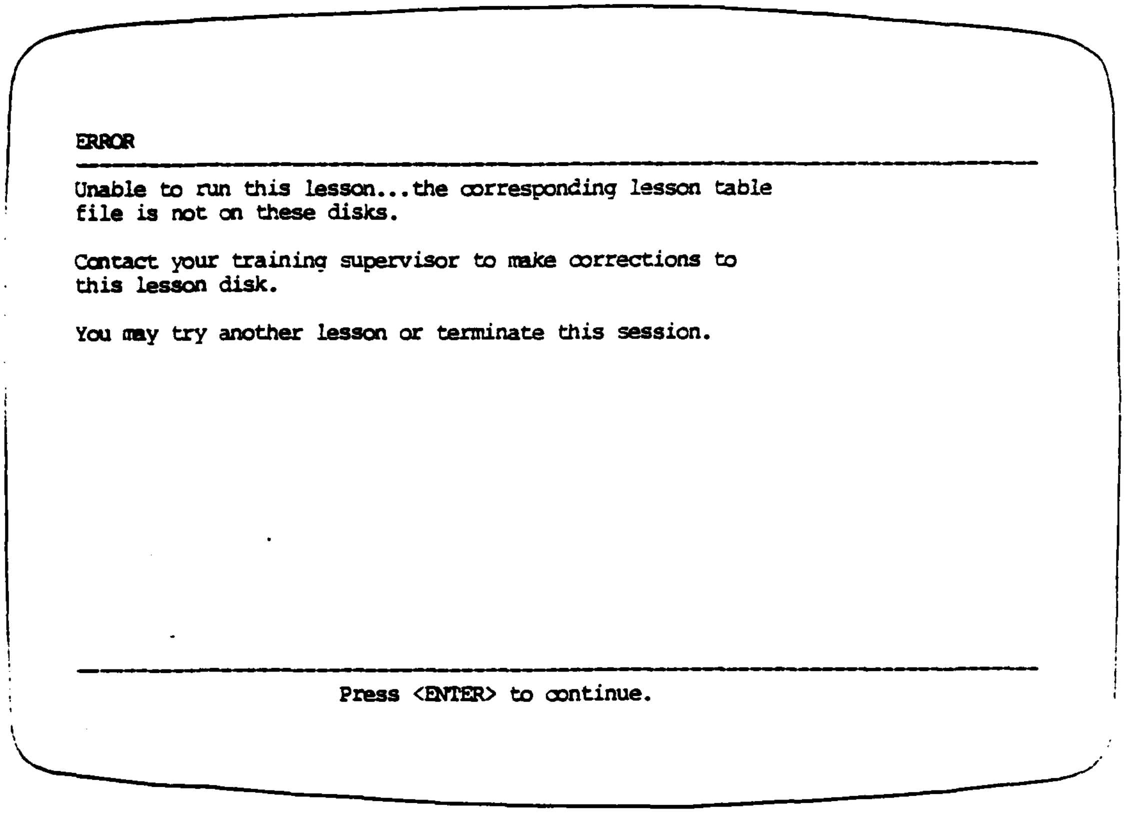
Figure B.1: LEARNER/BAS Courseware Example (continued)



```
MAIN MENU
------------------------------------------------------------------------
The following lesson files are available on this disk:

Drive :1  DATADISK   40 Cyl, DDEN, Free =  172.50K /   180.00K,  Date 13-Jul-87

LESSON1/TXT +

Enter lesson to run or change data disk and press <ENTER>:









------------------------------------------------------------------------
              Enter requested data and press <ENTER> to continue.
```

Display of the lesson catalogue. This screen is generated by displaying the files on the disk with an extension of "/TXT" or ".TXT". The student may enter the lesson filename to run, change the disk and press <ENTER> to display a new catalogue, of enter <Q><ENTER> to end program execution.

Figure B.1: LEARNER/BAS Courseware Example (continued)



ERROR

Unable to run this lesson...the corresponding lesson table file is not on these disks.

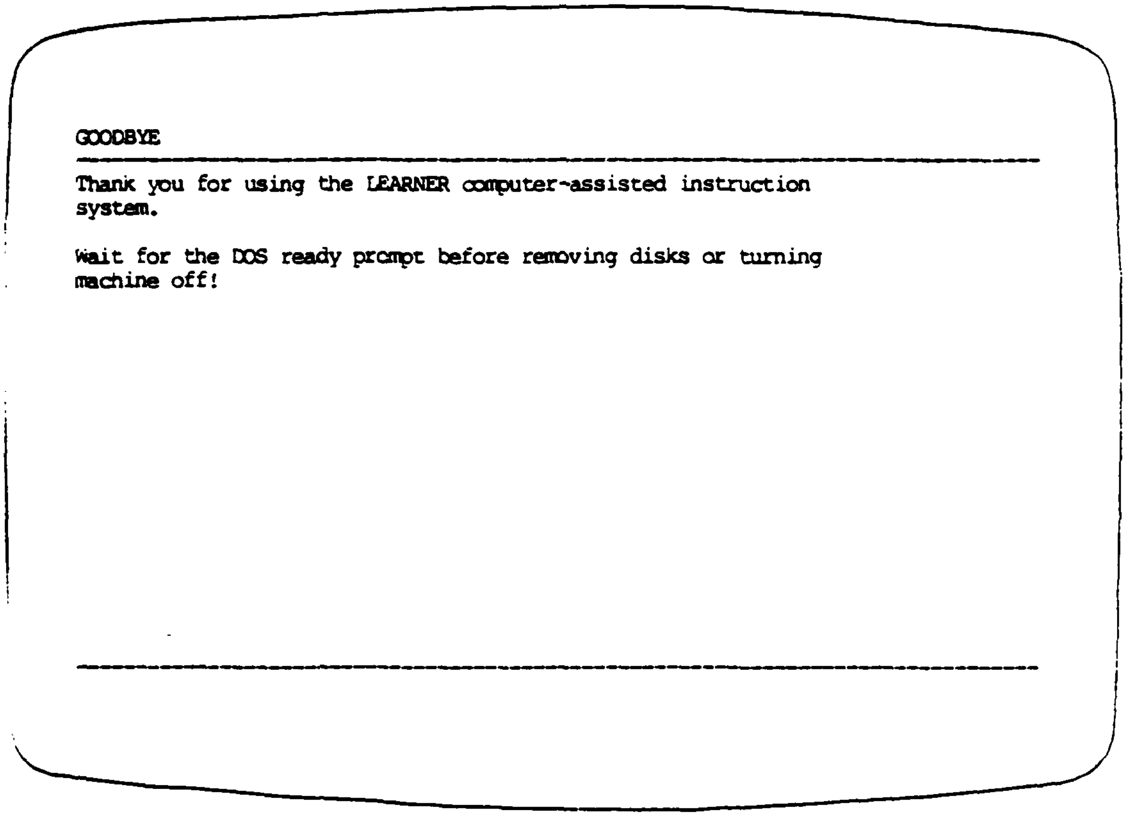
Contact your training supervisor to make corrections to this lesson disk.

You may try another lesson or terminate this session.

Press <ENTER> to continue.

Typical error message following erroneous input of lesson name. Pressing <ENTER> at this point returns the program to the main menu/lesson catalogue display.

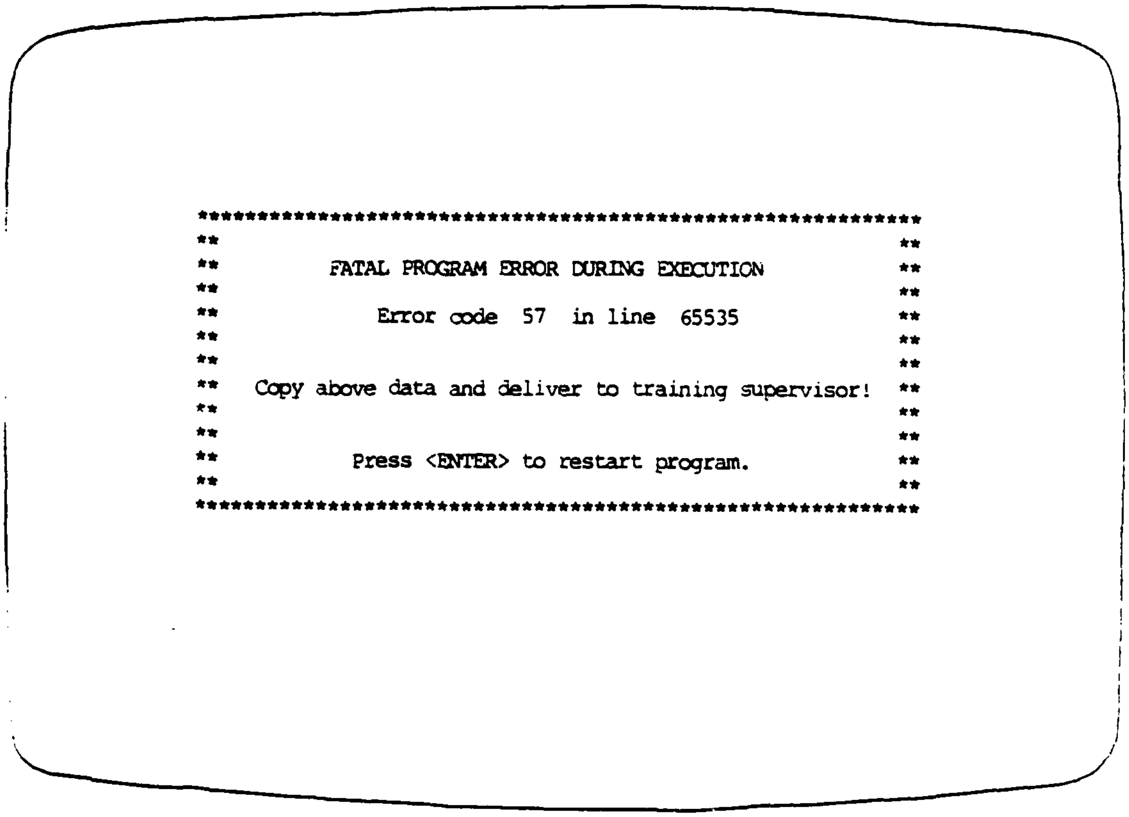
Figure B.1: LEARNER/BAS Courseware Example (continued)



```
GOODBYE
------------------------------------------------------------------------
Thank you for using the LEARNER computer-assisted instruction
system.

Wait for the DOS ready prompt before removing disks or turning
machine off!









------------------------------------------------------------------------
```

Screen display following a <Q><ENTER> input at the main menu/lesson catalogue display. This message is displayed for a short period while applicable disk files are closed then exits BASIC and returns to the DOS ready prompt.

Figure B.1: LEARNER/BAS Courseware Example (continued)



```
****************************************************************
**                                                            **
**          FATAL PROGRAM ERROR DURING EXECUTION              **
**                                                            **
**              Error code  57  in line  65535                **
**                                                            **
**                                                            **
**   Copy above data and deliver to training supervisor!     **
**                                                            **
**                                                            **
**             Press <ENTER> to restart program.              **
**                                                            **
****************************************************************
```

Screen display following a fatal error during program execution. Pressing <ENTER> closes disk files, erases variables in memory, and returns to the main menu/lesson catalogue prompt.

Figure B.1: LEARNER/BAS Courseware Example (continued)

Appendix C: LEARNER/BAS User's Guide

Contents

Introduction

LEARNER/BAS is a program written in the BASIC programming language which is one component of the WRITE-LEARNER computer-assisted instruction system. LEARNER will administer interactive, computer-assisted instruction courseware developed with the WRITE/BAS program.

This user's guide is intended to provide the knowledge required to use the LEARNER program with a minimum of computer knowledge.

During lessons, the student will use three different types of screens - text screens, multiple choice question screens, and true/false question screens. By using the various screens, the student should increase his knowledge of the subject area.

Notes for Courseware Administrators

This user's guide is primarily for use with a Tandy-Radio Shack Model IV or Zenith Z-248 microcomputer system. With the wide diversity of MS-DOS and TRSDOS computers available and the diversity of operating systems and disk-operating system "shells" in use, an "all-encompassing" user's guide for this system is not practible. This guide is being copied (as written) in ASCII format on the WRITE-LEARNER distribution disk. Courseware administrators will use this guide to develop local instructions depending on computers available and operating systems in use. Also, the use of job control language files (Model IV) or auto-batch command files (MS-DOS) to start the program is encouraged.

The user may need some BASIC programming knowledge depending on local computer configuration. The LEARNER program is configured to operate with the system disk on a disk drive designated drive 1 (TRS-DOS) or A (MS-DOS). If the local computer configuration does not permit this, lines 11215, 11220, and 12065 will require modification to include the correct drive designation.

Getting Started

LEARNER operates under the BASIC programming language using data disks from the WRITE-LEARNER package and locally developed courseware data disks. To start a lesson, take the following actions:

Step 1 Turn on the computer and monitor.

Step 2 Load BASIC into the computer. On Air Force Institute of Technology (AFIT) Z-248's, this is accomplished by pressing the <B> key while at the main menu display. On the Model IV, type "BASIC <ENTER>" at the DOS ready prompt.

Step 3 Insert the WRITE-LEARNER system disk in drive A (Z-248) or drive 1 (Model IV).

Step 4 Depending on computer configuration, type one of the following lines exactly as it appears then press the <ENTER> key:

```
RUN "A:LEARNER.BAS" (Z-248)
RUN "LEARNER/BAS:1" (Model IV)
```

Step 5 The LEARNER main menu will be displayed at this point. Ensure that the <CAPS LOCK> key is depressed. Enter your name as instructed by the program. Enter your first initial and last name with no space and press the <ENTER> key. If the <CAPS LOCK> key has not been depressed, the program will respond with an error message.

Step 6 After entering your name, the program will ask if you want to run a short introductory lesson about the LEARNER computer-assisted instruction program before starting a lesson. Respond to this question by pressing the <Y> or <N> key. Do not press the <ENTER> key after your response. Press only the <Y> or <N> key.

Step 7 If you respond to the introductory lesson question with <Y>, a short lesson will be run to give practice with the program. Follow the lesson prompts exactly - this program will not let you damage the computer, lessons, or disks. After the introductory lesson, the program will display the lesson catalogue as if you had pressed <N>. You may continue with step 8 from this point.

Step 8 If you respond to the introductory lesson question with <N>, the program will display the lesson files available on the disk in drive A (Z-248) or drive 1 (Model IV). If you do not see a lesson that you want to run, change the disk and press the <ENTER> key - a new lesson catalogue will be displayed. If a lesson is on the disk that you want to take, enter the lesson name (up to eight characters exactly as they appear on the catalogue) and press the <ENTER> key. The lesson will start after a short loading period.

Step 9 If you desire to terminate a session from an "Enter lesson to run or change data disk and press <ENTER>" prompt, enter the letter Q as the lesson name and press <ENTER>. The program will quit the session and return to the DOS ready prompt or a SHELL main menu display.

That's all the prerequisite knowledge required to operate the LEARNER program. The computer will prompt at virtually every step. Just take the action described on the instruction line of the display and you can't go wrong!

Stopping Sessions

You should complete each lesson started. At the end of a lesson, the program will return to a lesson catalogue prompt. Again, enter the letter Q at this prompt and press <ENTER> to terminate the program. At any point in the lesson execution, you can press the <CTRL> key and <E> key together to terminate the lesson early. However, if you stop the session early, your results will not be recorded.

Error Messages

As with any computer program, every precaution has been taken to ensure that the program is error free; however, unforeseen errors may occur. In the event that an error occurs, an error message will appear on the screen. You should copy the data as displayed and deliver it to your training supervisor for resolution. By pressing <ENTER> at the error message, the program will be restarted.

Appendix D: WRITE/BAS Program Documentation

Contents

WRITE/BAS Program Overview

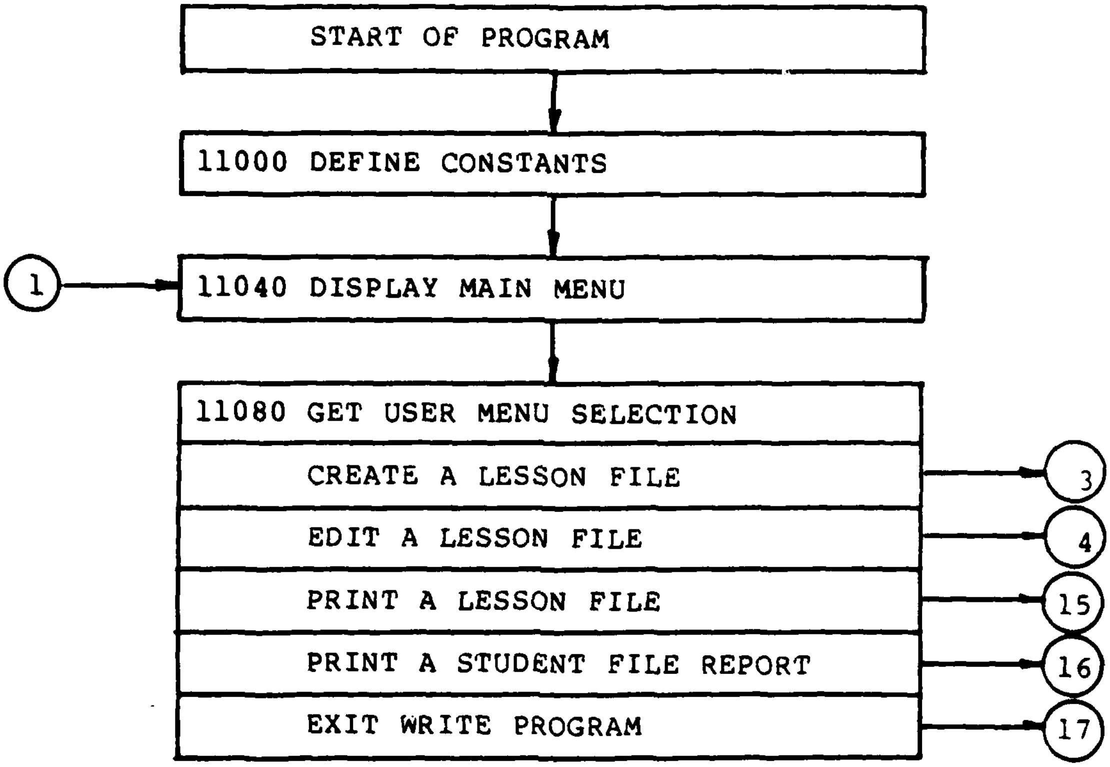
WRITE/BAS is a program written in the BASIC programming language which is one component of the WRITE-LEARNER computer-assisted instruction system. The system was developed on a Tandy/Radio Shack Model IV microcomputer and subsequently converted and tested on a Zenith Z-248 (IBM PC compatible) microcomputer. WRITE/BAS assists in the development of interactive, computer-assisted instruction courseware modules for use by the LEARNER/BAS program.

This appendix contains the documentation for the WRITE/BAS program. Included are comments about program operation; block diagrams to illustrate the program logic; a discription of the variables used in the program; cross-reference listings by variable name, line number, and BASIC keywords; and the program listings (TRSDOS and MSDOS versions). A user's guide for the program is included as a separate appendix.

WRITE/BAS is essentially a specialized data-base manager with limited word processing capabilities. Lesson courseware is stored in formatted, direct-access, disk records which are written as required during lesson development. A header records contains certain data about a lesson page (page type, page number, and jump pages). Following the header record are from 1 to 20 text records which comprise the lesson page. A second file, a sequential ASCII lesson table file, is generated from the lesson text file for reading into memory during lesson execution or lesson file editing to allow improved response times in locating records in the text file. A third file, a sequential ASCII student file, is generated during lesson execution and stores student data (student name, date lesson executed, and total/correct/incorrect question responses) about lesson use. WRITE/BAS uses this last file to produce hardcopy student file reports for use by courseware developers. Refer to the WRITE/BAS Record File Formats for a description of these files.

Figure D.1 is a simplified block diagram illustrating the operation of WRITE/BAS. Line numbers refer to both version of the program (i.e., TRSDOS and MSDOS) since line number consistency was maintained during program conversion.

If attempting to compile this program or convert it to operate on another type of computer, several cautions are in order. First, the program uses REMARK line numbers in program flow. GOTO and GOSUB commands must be changed to reflect deletions of REMARK statements. Second, the MSDOS version of WRITE/BAS emulates certain functions of the TRSDOS version (PRINT CHR$(30) in TRSDOS BASIC erases display to end of current line which was emulated in MSDOS BASIC by LOCATE xx,yy:PRINT STRING$(81-POS(0),32);:LOCATE xx,yy). Conversion should begin using the TRSDOS version program listing since this was the original program developed and the cross-reference listings included in this appendix refer to this version.



NOTE: → (n) indicates branch to corresponding (n) →.

Figure D.1: WRITE/BAS Block Diagram

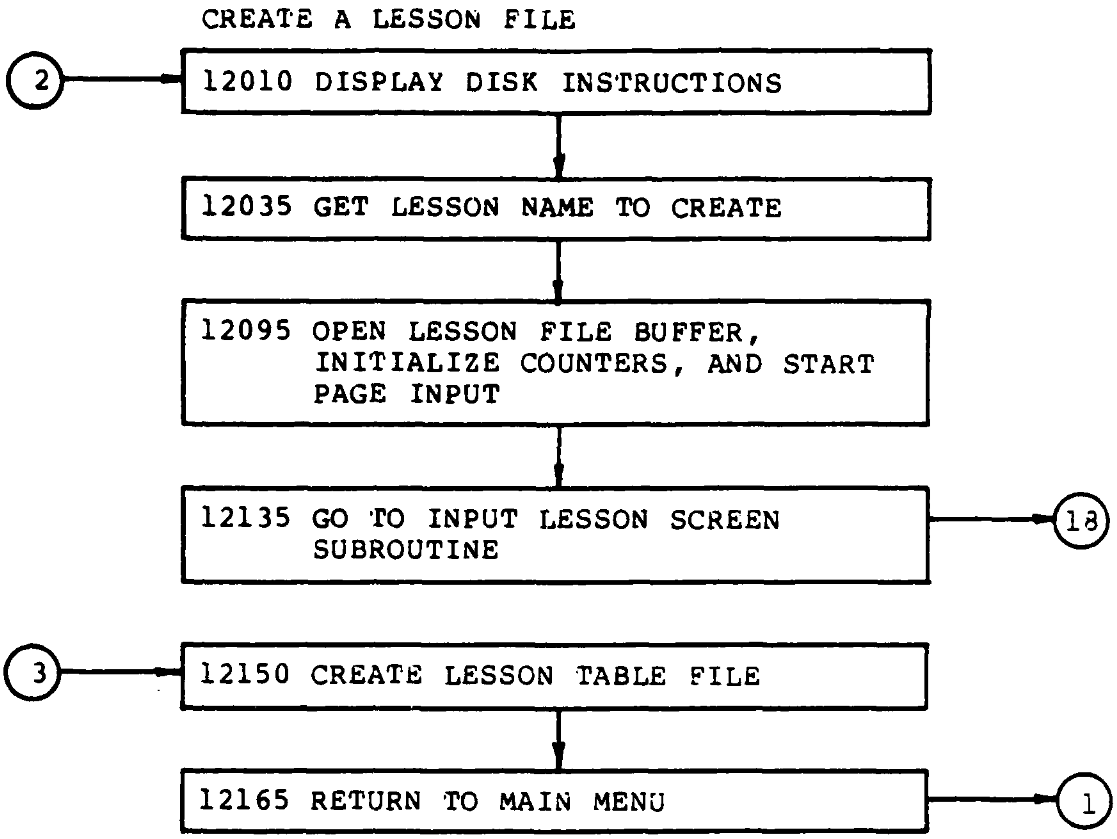

CREATE A LESSON FILE
2
12010 DISPLAY DISK INSTRUCTIONS
12035 GET LESSON NAME TO CREATE
12095 OPEN LESSON FILE BUFFER, INITIALIZE COUNTERS, AND START PAGE INPUT
12135 GO TO INPUT LESSON SCREEN SUBROUTINE
18
3
12150 CREATE LESSON TABLE FILE
12165 RETURN TO MAIN MENU
1

Figure D.1: WRITE/BAS Block Diagram (continued)

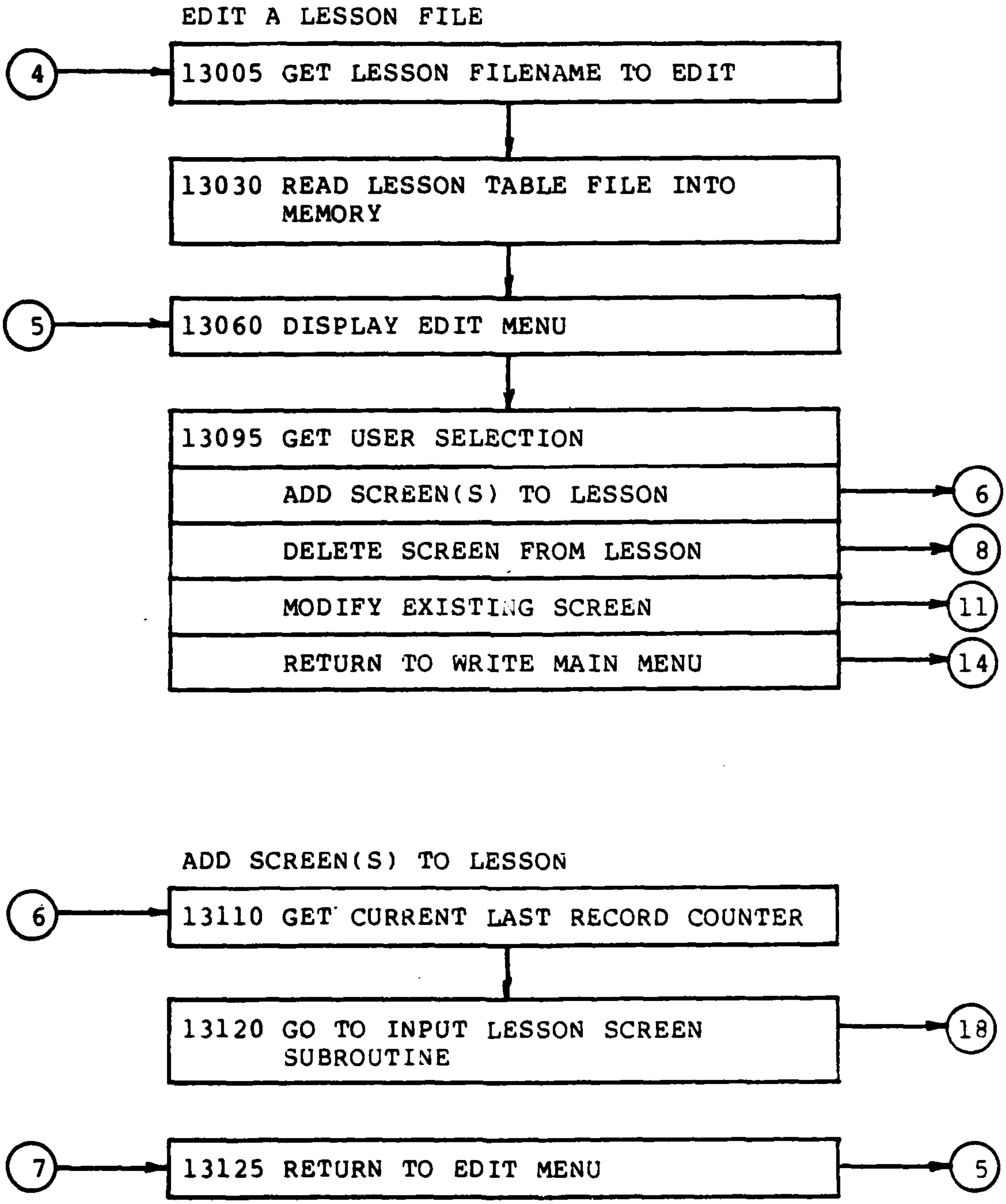

EDIT A LESSON FILE
4
13005 GET LESSON FILENAME TO EDIT
13030 READ LESSON TABLE FILE INTO MEMORY
5
13060 DISPLAY EDIT MENU
13095 GET USER SELECTION
ADD SCREEN(S) TO LESSON
6
DELETE SCREEN FROM LESSON
8
MODIFY EXISTING SCREEN
11
RETURN TO WRITE MAIN MENU
14
ADD SCREEN(S) TO LESSON
6
13110 GET CURRENT LAST RECORD COUNTER
13120 GO TO INPUT LESSON SCREEN SUBROUTINE
18
7
13125 RETURN TO EDIT MENU
5

Figure D.1: WRITE/BAS Block Diagram (continued)

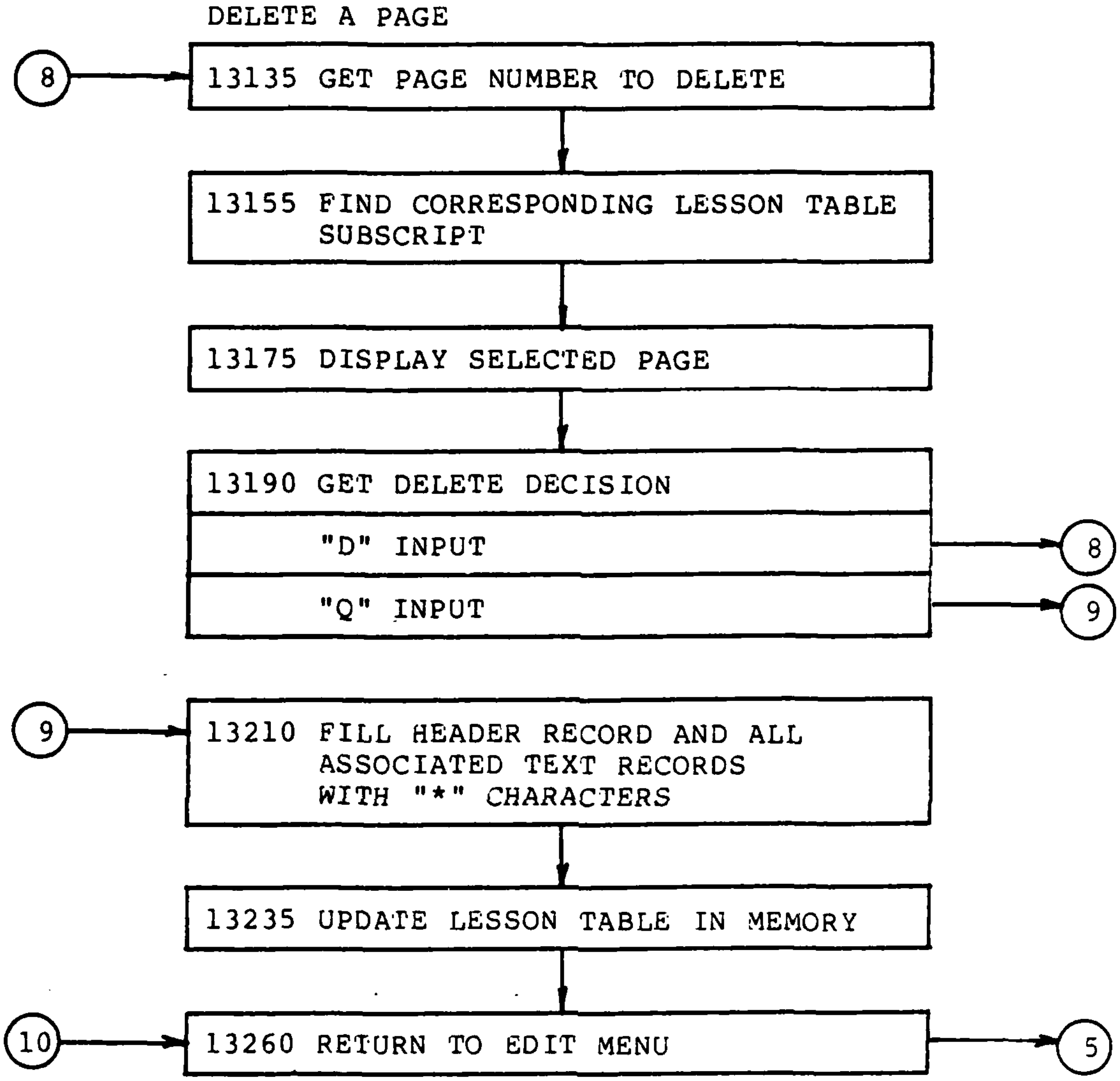

DELETE A PAGE
8
13135 GET PAGE NUMBER TO DELETE
13155 FIND CORRESPONDING LESSON TABLE SUBSCRIPT
13175 DISPLAY SELECTED PAGE
13190 GET DELETE DECISION
"D" INPUT
8
"Q" INPUT
9
9
13210 FILL HEADER RECORD AND ALL ASSOCIATED TEXT RECORDS WITH "*" CHARACTERS
13235 UPDATE LESSON TABLE IN MEMORY
10
13260 RETURN TO EDIT MENU
5

Figure D.1: WRITE/BAS Block Diagram (continued)

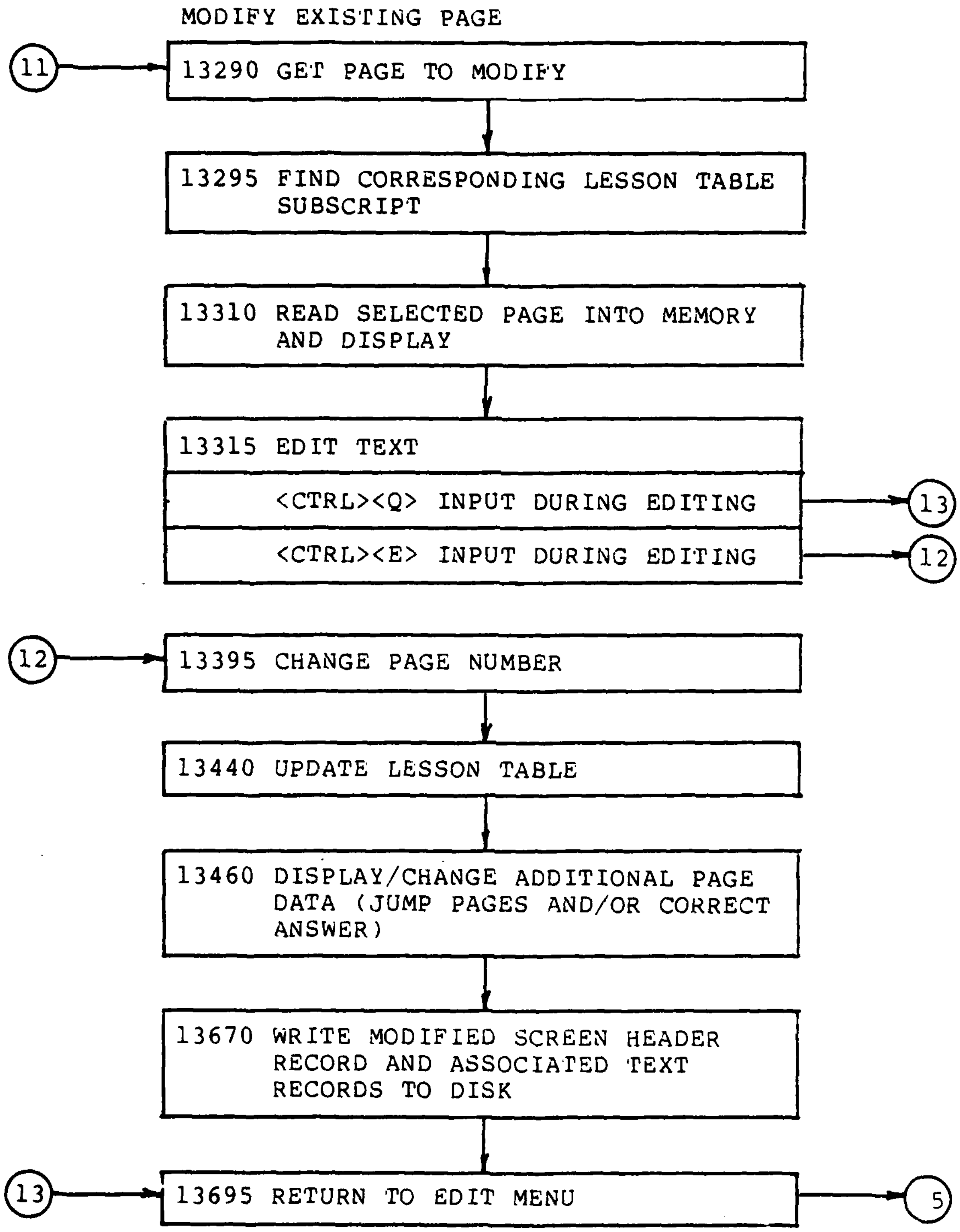

MODIFY EXISTING PAGE
11
13290 GET PAGE TO MODIFY
13295 FIND CORRESPONDING LESSON TABLE SUBSCRIPT
13310 READ SELECTED PAGE INTO MEMORY AND DISPLAY
13315 EDIT TEXT
<CTRL><Q> INPUT DURING EDITING
13
<CTRL><E> INPUT DURING EDITING
12
12
13395 CHANGE PAGE NUMBER
13440 UPDATE LESSON TABLE
13460 DISPLAY/CHANGE ADDITIONAL PAGE DATA (JUMP PAGES AND/OR CORRECT ANSWER)
13670 WRITE MODIFIED SCREEN HEADER RECORD AND ASSOCIATED TEXT RECORDS TO DISK
13
13695 RETURN TO EDIT MENU
5

Figure D.1: WRITE/BAS Block Diagram (continued)

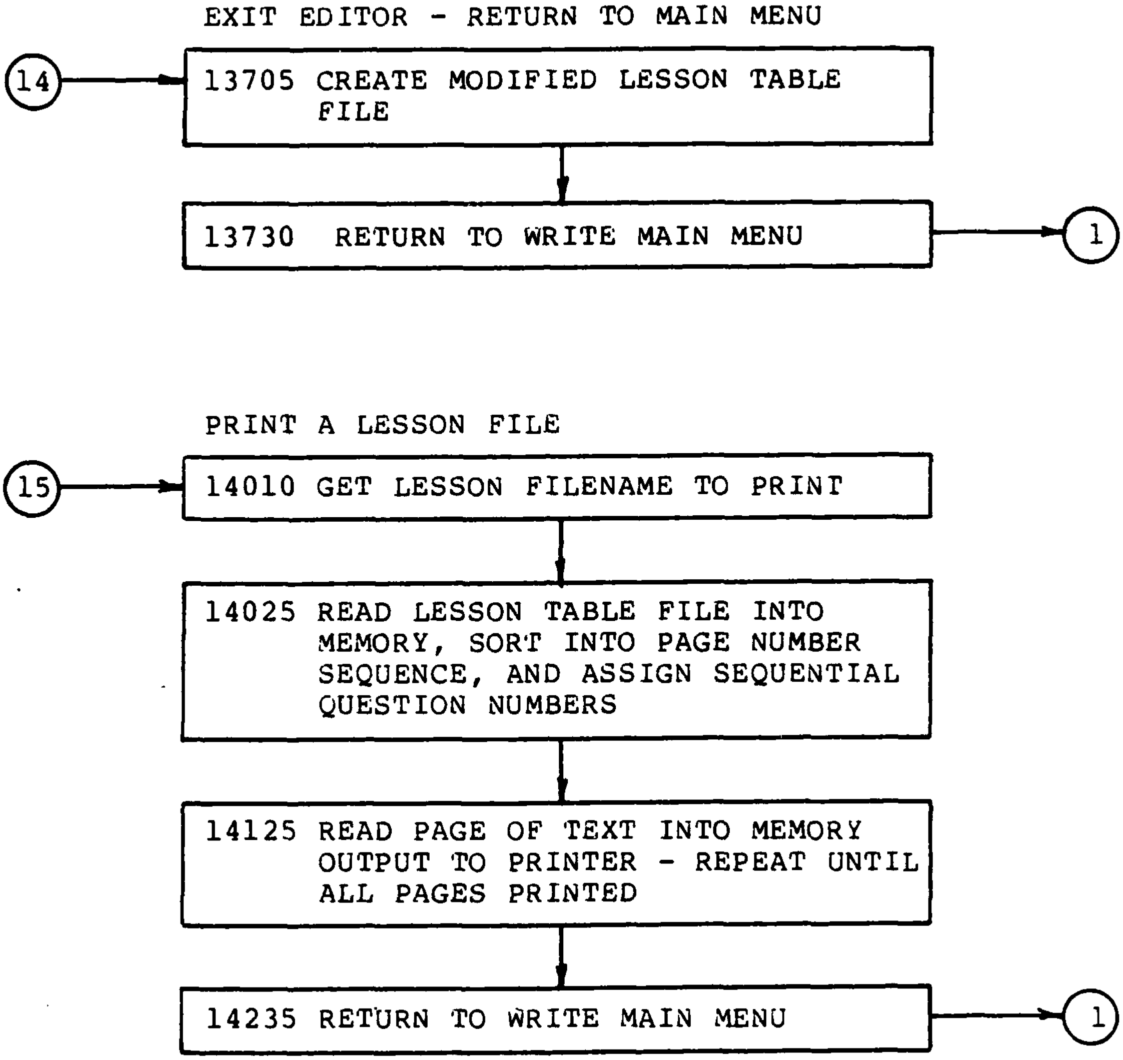

EXIT EDITOR - RETURN TO MAIN MENU
14
13705 CREATE MODIFIED LESSON TABLE FILE
13730 RETURN TO WRITE MAIN MENU
1
PRINT A LESSON FILE
15
14010 GET LESSON FILENAME TO PRINT
14025 READ LESSON TABLE FILE INTO MEMORY, SORT INTO PAGE NUMBER SEQUENCE, AND ASSIGN SEQUENTIAL QUESTION NUMBERS
14125 READ PAGE OF TEXT INTO MEMORY OUTPUT TO PRINTER - REPEAT UNTIL ALL PAGES PRINTED
14235 RETURN TO WRITE MAIN MENU
1

Figure D.1: WRITE/BAS Block Diagram (continued)

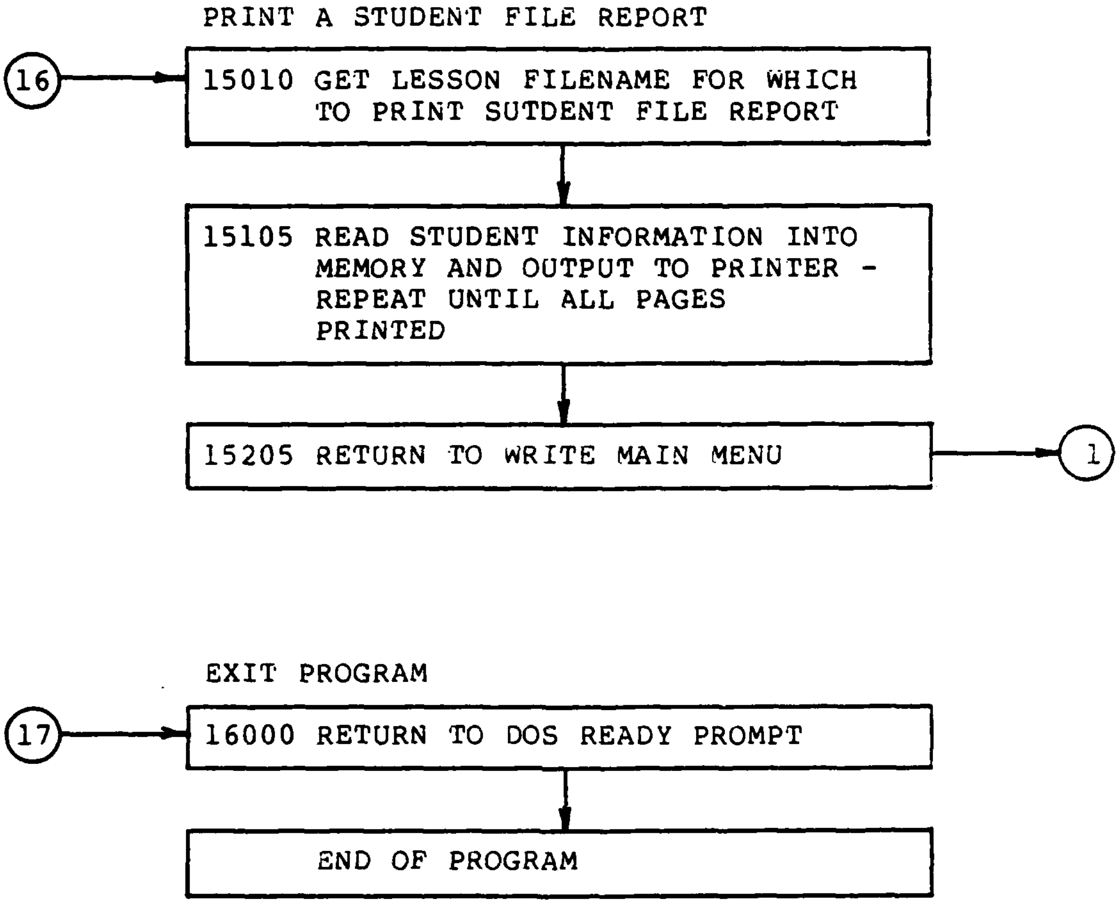

PRINT A STUDENT FILE REPORT
16
15010 GET LESSON FILENAME FOR WHICH TO PRINT SUTDENT FILE REPORT
15105 READ STUDENT INFORMATION INTO MEMORY AND OUTPUT TO PRINTER - REPEAT UNTIL ALL PAGES PRINTED
15205 RETURN TO WRITE MAIN MENU
1
EXIT PROGRAM
17
16000 RETURN TO DOS READY PROMPT
END OF PROGRAM

Figure D.1: WRITE/BAS Block Diagram (continued)

INPUT LESSON SCREEN SUBROUTINE

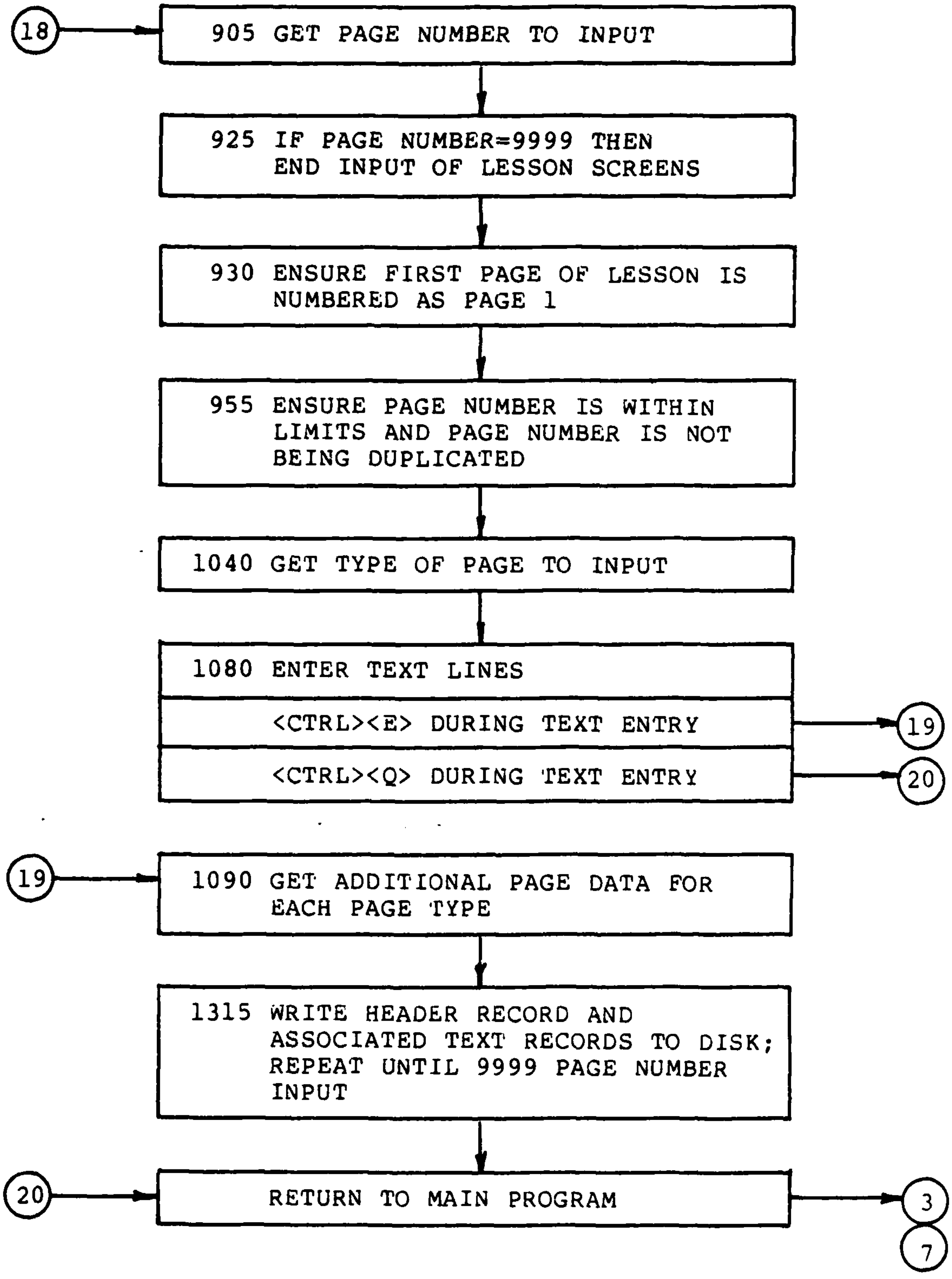

18
905 GET PAGE NUMBER TO INPUT
925 IF PAGE NUMBER=9999 THEN END INPUT OF LESSON SCREENS
930 ENSURE FIRST PAGE OF LESSON IS NUMBERED AS PAGE 1
955 ENSURE PAGE NUMBER IS WITHIN LIMITS AND PAGE NUMBER IS NOT BEING DUPLICATED
1040 GET TYPE OF PAGE TO INPUT
1080 ENTER TEXT LINES
<CTRL><E> DURING TEXT ENTRY
19
<CTRL><Q> DURING TEXT ENTRY
20
19
1090 GET ADDITIONAL PAGE DATA FOR EACH PAGE TYPE
1315 WRITE HEADER RECORD AND ASSOCIATED TEXT RECORDS TO DISK; REPEAT UNTIL 9999 PAGE NUMBER INPUT
20
RETURN TO MAIN PROGRAM
3
7

Figure D.1: WRITE/BAS Block Diagram (continued)

WRITE/BAS Variable List

A$	String for user input from keyboard.
AJUMP%	Go to lesson page number for <A>/<T>/<ENTER> response.
AJUMP	Temporary input value for AJUMP%.
ANSWER$	Correct answer for question (A-E or T/F).
BJUMP%	Go to lesson page number for <B>/<F> response.
BJUMP	Temporary input value for BJUMP%.
BUFi$	Temporary buffer variable for use with direct access files (i=integer value).
CC%	Cursor column position counter.
CJUMP%	Go to lesson page number for <C> response.
CJUMP	Temporary input value for CJUMP%.
COUNT	Counter for determining file record numbers.
DASH$	String of 80 dash characters.
DJUMP%	Go to lesson page number for <D> response.
DJUMP	Temporary input value for DJUMP%.
DUMMY$	Dummy variable for reading unused portions of direct access files.
EJUMP%	Go to lesson page number for <E> response.
EJUMP	Temporary input value for EJUMP%.
ENTER$	"Press <ENTER>..." instruction line.
FLAG%	Flag variale returned from subroutines to identify specific disk errors.
I	FOR/NEXT loop counter.
J	FOR/NEXT loop counter.
LASTREC	Number of last record in a direct-access file.
LC%	Cursor row position counter.
LEGAL$	String containing all allowable letter key inputs.
MCQP$	"Multiple Choice Question Page"
MORE%	Number of associated disk records required to generate lesson page in lesson text file.
MORE%(	Number of associated disk records required to generate lesson page in lesson table array.
NEED$	String containing codes for files required by program modules.
NUMPAGES%	Number of pages in lesson text file.
PAGE%	Lesson page number in lesson text file.

PAGE%(	Lesson page number in lesson in lesson table array.
PAGE	Temporary input value for PAGE%.
PCT	Percentage of correct question responses.
PROMPT$	"Press <letter> of your choice..." instruction line.
PTEST%	Temporary page number value for determining if page number is within legal limits.
QDATA$	"Enter requested data and press <ENTER>..." instruction line.
QDATE$	String containing date student performed lesson.
QEDIT$	Text editor instruction line.
QINST$	Instruction line to be displayed on lesson screen.
QMAT%	Array subscript in lesson table.
QNAME$	Lesson name to be displayed on lesson screen.
QNUM%(	Question number in lesson table.
QNUM%	Temporary counter for assigning question numbers.
QPAGE$	Page number to be displayed on lesson screen.
QPAGE%	Lesson page number being displayed during lesson execution; active page number.
QTEST$	File name for determining if file is on disk.
QTYPE$	Page type to be displayed during lesson development.
RCOUNT%	Counter for number of correct responses entered.
REC	Record number for direct acces file input/output.
STAR$	String of 60 asterisk characters.
START%	Starting lesson text file record number in lesson text file.
START%(	Starting lesson text file record number in lesson table.
STUDENT$	Student's name (first initital and last name; no space).
STUFILE$	Name of file for writing student lesson data.
TABLE$	Name of file for lesson table.
TCOUNT%	Counter for total number of questions asked.
TEST$	Temporary string for determining legal word-wrap position.
TEST%	Temporary value for determining legal word-wrap position.
TEXT$	Name of disk file for lesson text.

TEXT$(	Array holding text lines for lesson page.
TFQP$	"True/False Question Page"
TNAME$	Lesson name being used.
TP$	"Text Page"
TPOS	Temporary cursor column position.
TROW	Temporary cursor row position.
TYPE$	Lesson page type indicator (#=text page; ?=question page).
TYPE$(	Lesson page type indicator in lesson table.
WCOUNT%	Counter for number of incorrect question responses.
WQUEST$(	Array to hold wrong question responses.
WQUEST%	Array to hold wrong question numbers.
WQUEST%(	Array to hold wrong question numbers.
YN$	"<Y>es <N>o..." instruction line.

WRITE/BAS Record File Foımats

WRITE/BAS produces three files for use by the LEARNER/BAS program and uses one file produced by the WRITE/BAS program. The following pages contain the record file formats for these files. Although sequential ASCII files are not formatted (data elements are variable length), lengths are provided for these data elements for reference purposes.

The first file produced by WRITE/BAS, <Lesson Name>/TXT is comprised of two different types of formatted, direct-access records which form a page "block". A header record containing information about the lesson page is followed by from one to twenty text records which contain the page text material.

The second file, <Lesson Name>/TAB, is a sequential ASCII file which contains certain information from the text header records. This file is loaded into memory permitting very rapid access of the proper text records.

The third file, <Lesson Name>/STU, is a sequential ASCII file which contains data generated during lesson use. This file is appended on completion of each execution of a lesson module. If the file does not exist then it is created by the LEARNER/BAS program.

File: <Lesson Name>/TXT (Header records)
Type: Formatted, direct access

Length	Variable/s	Type	Remarks
1	TYPE$ BUF1$	A	Page type; #=text page, ?=question page, null/blank= text record, *=deleted record.
5	PAGE% BUF2$	N	User defined page number; used for lesson branching/reference purposes.
5	AJUMP% BUF3$	N	Branch to page for <A> response to multiple choice question, <T> response to true/false question, or <ENTER> key for text page.
5	BJUMP% BUF4$	N	Branch to page for <B> response to multiple choice question or <F> response to true/false question.
5	CJUMP% BUF5$	N	Branch to page for <C> response to multiple choice question.
5	DJUMP% BUF6$	N	Branch to page for <D> response to multiple choice question.
5	EJUMP% BUF7$	N	Branch to page for <E> response to multiple choice question.
1	ANSWER$ BUF8$	A	Correct response for question; must equal A-E or T/F.
3	MORE% BUF9$	N	Number of additional records of page (i.e., number of text lines); must equal 1-20 (third character reserved by BASIC for sign).
46	DUMMY$		Not used in header records.

Total record length = 81

File: <Lesson Name>/TXT (text records)
Type: Formatted, direct access

Length	Variable/s	Type	Remarks
1	-	A	Null/blank=text record; *=deleted record.
80	TEXT$	A/N	Text line.

Total record length = 81

File: <Lesson Name>/TAB
Type: Sequential ASCII

Length	Variable/s	Type	Remarks
5	PAGE% BUF2$	N	User defined page number; used for lesson branching.
4	START%	N	Starting file record number for PAGE% header record in <Lesson Name>/TXT file.
4	MORE%	N	Number of text records in <Lesson Name>/TXT file comprising PAGE%.
1	TYPE$	A	Page type

File: <Lesson Name>/STU
Type: Sequential ASCII

Length	Variable/s	Type	Remarks
NA	STUDENT$	A	Student name (first initial and last name; no space).
NA	DATE$	A/N	Variable length; format as returned by applicable BASIC/DOS.
4	TCOUNT%	N	Total questions asked during lesson (may differ from total questions in a lesson due to branching and/or repeated questions).
4	RCOUNT%	N	Number of correct responses.
4	WCOUNT%	N	Number of incorrect responses.
4	WQUEST%(i)	N	Question number answered incorrectly (i=1 to WCOUNT%).
1	WQUEST$(i)	A	Incorrect response to question (i=1 to WCOUNT).

WRITE/BAS BASIC Variables Cross-Reference List

```
A$
           155    160    165    170    175    200    205
           210    260    265    270    305    320    340
           340    375    395    415    415    420    425
           455    455    460    505    545    545    550
          1060   1065   1070   1160   1160   1265   1265
         11085  11085  11085  11085  11085  13100  13100
         13100  13100  13205  13335  13340  13345  13350
         13355  13360  13365  13370  13375  13380  13380
         13380  13380  13415  13500  13530  13535  13535
         13555  13565  13570  13575  13580  13585  13615
         13620  13620  13640  13655  13660  17045  17045
AJUMP%
           830   1135   1135   1170   1170   1275   1275
          1335  13490  13500  13545  13565  13630  13655
         13675  14185  14190  14195
AJUMP
          1135   1135   1135   1170   1170   1170   1275
          1275   1275  13500  13500  13500  13500  13565
         13565  13565  13565  13655  13655  13655  13655
ANSWER$
           855   1160   1265   1360  13465  13465  13520
         13535  13605  13620  13675  14190  14190  14195
         14195
BASE
         10025
BJUMP%
           835   1185   1185   1290   1290   1340  13545
         13570  13630  13660  13675  14190  14195
BJUMP
          1185   1185   1185   1290   1290   1290  13570
         13570  13570  13570  13660  13660  13660  13660
BUF
           570    570    570    570    570    570    570
           570    570    595    595    605    615    620
           825    830    835    840    845    850    855
           860    875    880   1325   1330   1335   1340
          1345   1350   1355   1360   1365   1405   1410
         12115  12115  12115  12115  12115  12115  12115
         12115  12115  12120  12120  13045  13045  13045
         13045  13045  13045  13045  13045  13045  13050
         13050  13215  13220  13675  13675  13675  13675
         13675  13675  13675  13675  13675  13690  13690
         14110  14110  14110  14110  14110  14110  14110
         14110  14110  14115  14115
CC%
           250    255    275    280    310    315    325
           325    345    355    355    380    430    430
           435    470    525
```

```
CJUMP%
           840    1200    1200    1345   13545   13575   13675
         14195
CJUMP
          1200    1200    1200   13575   13575   13575   13575
COUNT
           575     600     600     605     610     615     620
           640
DASH$
           130     135   10030   14140   14215   15060   15070
         15190
DJUMP%
           845    1215    1215    1350   13545   13580   13675
         14195
DJUMP
          1215    1215    1215   13580   13580   13580   13580
DUMMY$
           570     875    1370   12115   13045   13675   14110
EJUMP%
           850    1230    1230    1355   13545   13585   13675
         14195
EJUMP
          1230    1230    1230   13585   13585   13585   13585
ENTER$
           940     965    1010    1475    1595   10065   12015
         12185   13165   13305   13450   14075   15020
FLAG%
           230     230    1140    1175    1190    1205    1220
          1235    1280    1295
I
           585     590     625     640     645     645     645
           645     650     690     700     700     700     700
           705     705     720     730     730     740     745
           750     755     760     770     790     795     795
           800     865     870     880     885     985     990
           995    1155    1155    1155    1260    1260    1260
          1395    1410    1420    1520    1525    1530    1535
          1655    1660    1665   12060   12065   12070   12075
         13240   13245   13245   13245   13245   13245   13245
         13245   13245   13245   13400   13400   13400   13435
         13435   13435   13690   13690   13690   14045   14050
         14055   14065   15125   15130   15130   15135   15140
         15155   15155   15155   15165   17015   17015   17015
         17015
J
           735     740     745     750     755     760     765
         14165   14170   14175   15150   15155   15155   15155
         15160
LASTREC
           580     585
```

```
LC%
            250    255    275    275    285    285    290
            315    315    350    350    380    380    425
            425    450    465    465    470    470    475
            495    505    505    510    510    515    520
            520    530    530    550   1320
LEGAL$
            165   1055   1160   1265  11080  13095  13200
          13410  13495  13525  13535  13550  13560  13610
          13620  13635  13645
MCQP$
           1065  10050
MORE%
            860    865   1320   1365   1395   1440  12165
          13360  13380  13675  13690  13710  14165  14230
          17050
MORE%(
            615    645    700    755    755   1440   1655
          12165  13005  13210  13245  13245  14005
NEED$
           1570   1575   1580  13015  14020  15010
NUMPAGES%
            720    730    735    790    985   1000   1000
           1440   1440   1440   1440  11030  13240  13250
          13250  13435  14045  14130
PAGE%
            920    925    925    935    960    960    970
            990   1015   1035   1075   1330   1440  12165
          13150  13160  13180  13290  13300  13320  13405
          13440  13440  13675  13710  14230  17050
PAGE%(
            620    645    700    740    740    745    745
            795    990   1440  12005  13005  13245  13245
          13435  13440  14005  14150
PAGE
            920    920    920  13150  13150  13150  13290
          13290  13290  13430  13430  13435  13440  13445
PCT
          15110  15115
PROMPT$
           1035  10035  11045  13065  13535  13560  13620
          13645
PTEST%
            230    230   1135   1170   1185   1200   1215
           1230   1275   1290
QDATA$
            910   1095   1500  10040  12040  13140  13280
          13420  13500
QDATE$
          15105  15115
QEDIT$
           1075  10045
```

```
QINST$
            135     675     910     940     965    1010    1035
           1075    1475    1500    1595   11045   12015   12040
          12155   12185   13065   13140   13180   13280   13320
          13705   14075   15020
QMAT%
            795     805     820     870    1655   13165   13210
          13210   13210   13240   13305   13440   13680   14130
          14150   14155   14155   14185   14190   14195   14210
QNAME$
            130     675     910     940     965    1010    1035
           1075    1595   11045   12015   12040   12155   12185
          13005   13065   13140   13180   13280   13320   13705
          14015   14075   15005   15020
QNUM%(
          14055   14155
QNUM%
          14040   14055   14060   14060
QPAGE$
            130     675     910     940     965    1010    1035
           1075    1475    1500    1595   11045   12015   12040
          12155   12185   13065   13140   13180   13280   13320
          13705   14075   15020
QPAGE%
            795   13160   13300
QTEST$
           1570    1570    1575    1575    1580    1580    1600
QTYPE$
            130     675     910     940     965    1010    1035
           1060    1065    1070    1475    1500   11045   12015
          12040   12155   12185   13065   13140   13180   13280
          13320   13705   14075   15020
RCOUNT%
          15105   15110   15115
REC
            935     935    1375    1380    1400    1400    1415
           1425    1425   12130   13115   13210   13225   13230
          13680   13685   13690   13690   13690
STAR$
          17005   17015   17015
START%
           1375    1440   12165   13710   14230   17050
START%(
            610     645     700     750     750     820     870
           1440   12005   13005   13210   13210   13245   13245
          13680   14005
STUDENT$
          15105   15115
STUFILE$
           1560    1580   15090
TABLE$
            635     685    1555    1575   12085
```

```
TCOUNT%
           15105  15110  15115
TEST$
             495    500
TEST%
             170    175    175    175    490    495    500
             500    505    510    515    515    525   1525
            1530   1530   1530  12065  12070  12070  12070
TEXT$
             400    565   1450   1550   1570  12080  12110
           12165  13040  13205  13255  13345  13695  13710
           14105  14205  14230  17050
TEXT$(
             275    275    315    315    350    350    380
             380    425    425    465    465    495    505
             505    510    510    520    880   1410   1450
            1660  12005  13005  13205  13255  13345  13375
           13375  13375  13375  13380  13690  13695  14005
           14170  14205
TFQP$
            1070  10060
TNAME$
             675    910    940    965   1010   1035   1075
            1510   1515   1520   1525   1530   1550   1555
            1560   1610  12050  12055  12060  12065  12070
           12080  12085  12155  12195  13065  13140  13180
           13280  13320  14075  15020  15050
TP$
            1060  10055
TPOS
           13375  13375  13375  13375
TROW
           13375  13375  13375  13375  13375  13375  13375
TYPE$
             825   1060   1065   1070   1100   1105   1110
            1240   1300   1325   1440  12165  13465  13465
           13470  13475  13480  13640  13675  13710  14230
           17050
TYPE$(
             605    645    700    760    760   1440  12005
           13005  13245  13245  14005  14050  14155  14185
           14190  14195
WCOUNT%
           15105  15115  15120  15125  15140
WQUEST$(
           15095  15130  15155
WQUEST$
           15175  15200
WQUEST%
           15175  15200
WQUEST%(
           15095  15130  15155  15155
```

```
YN$
         10070  13405  13495  13525  13550  13610  13635
```

WRITE/BAS Line Number Cross-Reference List

```
  125 =>    675    910    940    965   1010   1035   1075
           1475   1500   1595  11045  12015  12040  12155
          12185  13065  13140  13180  13280  13320  13705
          14075  15020
  150 =>   1055   1160   1265  11080  13095  13200  13410
          13495  13525  13535  13550  13560  13610  13620
          13635  13645
  155 =>    165
  195 =>    210    950    975   1020   1490   1630  12025
          12205  13165  13305  13450  14090  15035
  225 =>   1135   1170   1185   1200   1215   1230   1275
           1290
  245 =>   1080
  255 =>    295    310    330    345    365    415    440
            480    535
  260 =>    265
  305 =>    270
  340 =>    305
  375 =>    340    550
  395 =>    375
  415 =>    395
  450 =>    435
  455 =>    455
  490 =>    460
  495 =>    500
  540 =>    290    450    475
  545 =>    545    550
  560 =>  12160  13715
  625 =>    595
  670 =>  13035  14030
  695 =>    710
  715 =>    695
  765 =>    740
  785 =>  13160  13300
  815 =>  13170  13310  14135
  900 =>  12135  13120
  905 =>    950    975   1020   1455
  915 =>    920
  955 =>    935
  980 =>    960
 1010 =>    990
 1030 =>   1005
 1115 =>   1100
 1120 =>   1140
 1130 =>   1135
 1150 =>   1105
 1165 =>   1170   1175
 1180 =>   1185   1190
 1195 =>   1200   1205
 1210 =>   1215   1220
 1225 =>  1230   1235
 1255 =>  1110
 1270 =>  1275   1280
 1285 =>  1290   1295
 1315 =>  1145   1245   1305
 1465 => 13020  14025  15010
 1500 =>  1635
 1590 =>  1565
 1620 =>  1600   1605   1610
 1625 =>  1620
 1650 => 13185  13325
10000 =>    10
11000 =>  1515  12055  17055
11025 => 12170  13730  14235  15205
12000 => 11085
12015 => 12215
12180 => 12105
12205 => 12190  12195
12215 => 12210
13000 => 11085
13060 => 13125  13165  13205  13260  13305  13345  13695
13105 => 13100
13130 => 13100
13145 => 13150
13270 => 13100
13285 => 13290
13335 => 13340  13355  13355  13360  13360  13365  13365
         13370  13370  13375  13380  13385
13390 => 13350
13405 => 13440  13455
13425 => 13430
13445 => 13435
13460 => 13415
13485 => 13470  13505
13500 => 13500
13510 => 13475
13520 => 13535
13540 => 13530  13590
13565 => 13565
13570 => 13570
13575 => 13575
13580 => 13580
13585 => 13585
13595 => 13480
13600 => 13620
13625 => 13615
13630 => 13665
13655 => 13655
13660 => 13660
13670 => 13500  13555  13640
13700 => 13100
14000 => 11085
14065 => 14050
```

```
14200 => 14185  14190
15000 => 11085
15095 => 15180
15170 => 15120
15185 => 15100
16000 => 11085
17000 =>  1515   1585   1640  10025  12125  12215
17045 => 17045
```

<u>WRITE/BAS BASIC Keywords Cross-Reference List</u>

```
*       15110
+         255     285     350     355     425     430     470
          505     505     515
          520     520     530     600     675     705     735
         870     1000    1400
         1425    1550    1555    1560   10035   10040   10045
        10045   10045   10045   10045   10055   10060   10065
        10070   12080   12085   12155   13115   13180   13180
        13180   13210   13245   13245   13245   13245   13360
        13360   13375   13375   13380   13690   13705   14060
        15155   15155   15155   17005
-         175     255     275     315     325     380     500
          505     525     720     730     805    1530   12070
        13165   13250   13305   13355   13355   13360   13365
        13375   13375   13375   13375   13375   13375   13375
        13380
/       15110
<         175     210     230     270     305     340     340
          375     395     415     460     515     550     960
         1530    1570    1575    1580   12070   13380   13415
        13500   13500   13530   13555   13565   13570   13575
        13580   13585   13615   13640   13655   13660   14050
        14190   14195   15155   17045
=         155     160     165     170     175     175     175
          200     205     230     230     250     250     260
          265     275     280     285     290     310     315
          325     350     355     380     425     430     435
          450     455     455     465     470     470     475
          490     495     500     500     505     510     525
          530     545     545     575     580     585     595
          595     600     605     610     615     620     640
          675     675     675     675     690     705     720
          730     735     790     795     795     805     825
          830     835     840     845     850     855     860
          865     875     880     910     910     910     910
          920     925     925     935     935     940     940
          940     940     960     960     965     965     965
          965     985     990    1000    1010    1010    1010
         1010    1035    1035    1035    1035    1055    1060
         1060    1060    1065    1065    1065    1070    1070
         1070    1075    1075    1075    1100    1105    1110
         1135    1135    1140    1155    1160    1160    1170
         1170    1175    1185    1185    1190    1200    1200
         1205    1215    1215    1220    1230    1230    1235
         1240    1260    1265    1265    1275    1275    1280
         1290    1290    1295    1300    1320    1325    1330
         1335    1340    1345    1350    1355    1360    1365
         1370    1375    1395    1400    1405    1410    1425
         1440    1440    1440    1440    1475    1475    1475
         1500    1500    1500    1515    1520    1525    1530
```

```
=          1530   1530   1550   1555   1560   1570   1575
           1580   1595   1595   1595   1600   1605   1610
           1655  10030  10035  10040  10045  10050  10055
          10060  10065  10070  11030  11045  11045  11045
          11045  11080  11085  11085  11085  11085  11085
          12015  12015  12015  12015  12040  12040  12040
          12040  12055  12060  12065  12070  12070  12070
          12080  12085  12130  12155  12155  12155  12155
          12185  12185  12185  12185  12190  12195  13005
          13020  13065  13065  13065  13065  13095  13100
          13100  13100  13100  13115  13140  13140  13140
          13140  13150  13160  13165  13180  13180  13180
          13180  13200  13205  13210  13215  13220  13240
          13245  13245  13245  13245  13250  13280  13280
          13280  13280  13290  13300  13305  13320  13320
          13320  13320  13335  13340  13345  13350  13355
          13355  13360  13360  13365  13365  13370  13370
          13375  13375  13375  13375  13380  13380  13380
          13380  13380  13400  13410  13435  13435  13440
          13440  13465  13465  13465  13465  13470  13475
          13480  13495  13500  13525  13535  13535  13550
          13560  13565  13565  13570  13570  13575  13575
          13580  13580  13585  13585  13610  13620  13620
          13635  13640  13645  13655  13655  13660  13660
          13675  13675  13675  13675  13675  13675  13675
          13675  13675  13675  13680  13690  13690  13690
          13690  13705  13705  13705  13705  14015  14020
          14040  14045  14055  14060  14075  14075  14075
          14075  14130  14155  14165  14185  14190  14195
          15005  15010  15020  15020  15020  15020  15110
          15120  15125  15140  15150  17005  17015  17045
>           175    210    230    270    290    305    340
            340    345    375    395    415    460    550
            740    920    935    960   1135   1170   1185
           1200   1215   1230   1275   1290   1530   1570
           1575   1580  12070  13150  13290  13380  13415
          13430  13500  13500  13530  13555  13565  13570
          13575  13580  13585  13615  13640  13655  13660
          14050  14190  14195  15155  17045
AND         175    340    935    960   1530  12070  13380
          13380  13465  13500  13565  13570  13575  13580
          13585  13655  13660  14190  14195
ASC         170    375    395    415    415    550   1525
          12065  13345  13350  13355  13360  13365  13370
          13375  13380  13380
CHR$        160    175    205    210    270    295    305
            340    340    460    515    540    915   1095
           1120   1125   1130   1155   1165   1180   1195
           1210   1225   1260   1270   1285   1530  12070
          13145  13165  13285  13305  13375  13380  13400
          13405  13405  13420  13425  13445  13450  13490
          13495  13500  13500  13520  13525  13535  13535
          13560  13560  13565  13570  13575  13580  13585
```

```
CHR$      13605  13610  13620  13620  13630  13645  13650
          13655  13660  14220  15195  17045
CLOSE       630    655    715   1515   1570   1575   1580
           1585   1640  12140  12165  12215  13040  13710
          13720  14225  15205  17055
CLS         130  12170  14235  15205  16010  17010
DATE$     15050
DIM        1450  12005  13005  13205  13255  13345  13695
          14005  14205  15095
ELSE        230    500    550    740    920   1135   1170
           1185   1200   1215   1230   1275   1290  11085
          11085  11085  11085  13100  13100  13100  13150
          13290  13355  13360  13365  13370  13435  13500
          13500  13565  13570  13575  13580  13585  13655
          13660  14155
END         160    205  16010  17060
EOF         695  15100
ERASE       400   1450  12165  13205  13255  13345  13695
          13725  14205  14230  15175  15200  17050
ERL       17025
ERR        1600   1605   1610  12190  12195  17025
ERROR      1515   1565   1585   1640  10025  12105  12125
          12215
FIELD       570  12115  12120  13045  13050  14110  14115
FOR         585    640    730    735    790    865    985
           1155   1260   1395   1520   1655  12060  13210
          13240  13400  13435  13690  14045  14130  14165
          15125  15140  15150  17015
GET         590    820    870
GOSUB       675    910    940    950    965    975   1010
           1020   1035   1055   1075   1080   1135   1160
           1170   1185   1200   1215   1230   1265   1275
           1290   1475   1490   1500   1595   1630  11045
          11080  12015  12025  12040  12135  12155  12160
          12185  12205  13020  13035  13065  13095  13120
          13140  13160  13165  13170  13180  13185  13200
          13280  13300  13305  13310  13320  13325  13410
          13450  13495  13525  13535  13550  13560  13610
          13620  13635  13645  13705  13715  14025  14030
          14075  14090  14135  15010  15020  15035
GOTO         10    295    330    365    440    480    500
            535    710    950    975   1005   1020   1145
           1245   1305   1455   1515   1515   1565   1585
           1600   1605   1610   1635   1640  10025  12055
          12105  12125  12170  12190  12195  12215  12215
          13125  13165  13205  13260  13305  13345  13350
          13355  13360  13365  13370  13375  13380  13385
          13440  13455  13500  13505  13535  13565  13570
          13575  13580  13585  13590  13620  13640  13655
          13660  13665  13695  13730  14185  14190  14235
          15120  15180  15205
IF          160    165    175    205    210    230    265
            270    290    305    310    340    345    375
```

```
IF          395    415    435    450    455    460    475
            500    515    545    550    595    695    740
            795    920    925    935    960    990   1060
           1065   1070   1100   1105   1110   1135   1140
           1170   1175   1185   1190   1200   1205   1215
           1220   1230   1235   1275   1280   1290   1295
           1515   1530   1570   1575   1580   1600   1605
           1610  11085  11085  11085  11085  11085  12055
          12070  12190  12195  13100  13100  13100  13100
          13150  13165  13205  13290  13305  13340  13345
          13350  13355  13355  13360  13360  13365  13365
          13370  13370  13375  13380  13380  13415  13430
          13435  13465  13470  13475  13480  13500  13500
          13530  13555  13565  13565  13570  13570  13575
          13575  13580  13580  13585  13585  13615  13640
          13655  13655  13660  13660  14050  14155  14185
          14190  14195  15100  15120  15155  17045
INKEY$      260    455    545  13335  17045
INPUT       155    200    700    920   1135   1170   1185
           1200   1215   1230   1275   1290   1510  12050
          13150  13290  13430  13500  13565  13570  13575
          13580  13585  13655  13660  15105  15130
INSTR       165    500   1570   1575   1580  14190  14195
LEFT$       275    315    380    465    510  13375
LEN        1520  12060
LOC         610
LOF         580  13115
LPRINT    14140  14145  14150  14155  14155  14160  14170
          14180  14185  14190  14195  14200  14215  14220
          15050  15055  15060  15065  15070  15075  15115
          15145  15155  15170  15170  15190  15195
LSET       1325   1330   1335   1340   1345   1350   1355
           1360   1365   1370   1405   1410  13215  13220
          13675  13675  13675  13675  13675  13675  13675
          13675  13675  13675  13690  13690
MID$        495   1525   1530  12065  12070  13380
NEXT        625    650    765    770    800    885    995
           1155   1260   1420   1535   1665  12075  13230
          13245  13400  13435  13690  14065  14175  14210
          15135  15160  15165  17015
ON         1515   1565   1585   1640  10025  12105  12125
          12215
OPEN        565    635    685   1570   1575   1580  12110
          13040  14105  15090
OPTION    10025
OR          230    415    595    925    935  13465
POS       13355  13360  13365  13365  13370  13370  13375
          13380  13380
PRINT       130    130    130    130    135    135    135
            255    295    320    360    420    515    520
            540    540    680    915    945    950    970
            975   1015   1020   1040   1045   1050   1055
           1095   1120   1125   1130   1155   1160   1160
```

```
PRINT        1165    1180    1795    1210    1225    1260    1265
             1265    1270    1285    1480    1485    1505    1505
             1600    1605    1610    1615    1625    1660   11050
            11055   11060   11065   11070   11075   12020   12025
            12045   12045   12190   12195   12200   12205   13070
            13075   13080   13085   13090   13145   13165   13165
            13165   13195   13285   13305   13305   13305   13330
            13355   13360   13365   13370   13375   13375   13380
            13380   13400   13405   13405   13405   13420   13425
            13445   13450   13450   13490   13495   13495   13500
            13500   13520   13525   13525   13535   13535   13535
            13545   13550   13550   13560   13560   13560   13560
            13565   13570   13575   13580   13585   13605   13610
            13610   13620   13620   13620   13620   13630   13635
            13635   13645   13645   13645   13650   13655   13660
            14080   14085   15025   15030   17015   17015   17015
            17015   17020   17025   17030   17035   17040
PUT          1380    1415   13225   13685   13690
REM             1       2       3       4       5       6       7
                8      10      15     100     105     110     115
              120     125     145     150     185     190     195
              220     225     240     245     260     270     300
              305     335     340     345     370     375     390
              395     410     415     445     460     485     555
              560     665     670     725     780     785     810
              815     895     900     905     930     955     980
             1025    1030    1085    1090    1115    1150    1250
             1255    1310    1315    1385    1390    1430    1435
             1445    1460    1465    1470    1495    1540    1545
             1590    1645    1650    1675   10000   10005   10010
            10015   10020   10075   11000   11005   11010   11015
            11020   11025   11035   11040   11090   12000   12010
            12030   12035   12090   12095   12100   12145   12150
            12170   12175   12180   13000   13010   13025   13030
            13055   13060   13105   13110   13130   13135   13155
            13175   13190   13235   13265   13270   13275   13295
            13315   13345   13350   13355   13360   13365   13370
            13375   13380   13385   13390   13395   13460   13485
            13510   13515   13540   13595   13600   13625   13670
            13700   13735   14000   14010   14035   14070   14095
            14100   14120   14125   14235   14240   15000   15015
            15040   15045   15080   15085   15185   15205   15210
            16000   16005   17000
RESUME       1620   12210   17055
RETURN        140     180     215     235     385     405     550
              660     775     795     805     890     925    1585
             1640    1670
RIGHT$        505   13375
ROW         13355   13355   13360   13360   13365   13370   13375
            13380   13380
STEP        15140
```

```
STR$          1035   1075   1330   1335   1340   1345   1350
              1355   1365  13180  13320  13675  13675  13675
             13675  13675  13675  13675
STRING$        350    360    675  10030  10035  10040  10045
             10045  10045  10055  10060  10065  10070  12155
             13180  13180  13220  13630  13705  15050  17005
             17005
SWAP           745    750    755    760
TAB(         14190  14195  15050  15145
THEN           160    165    175    205    210    230    265
               270    290    305    310    340    345    375
               395    415    435    450    455    460    475
               500    515    545    550    595    695    740
               795    920    925    935    960    990   1060
              1065   1070   1100   1105   1110   1135   1140
              1170   1175   1185   1190   1200   1205   1215
              1220   1230   1235   1275   1280   1290   1295
              1515   1530   1570   1575   1580   1600   1605
              1610  11085  11085  11085  11085  11085  12055
             12070  12190  12195  13100  13100  13100  13100
             13150  13165  13205  13290  13305  13340  13345
             13350  13355  13355  13360  13360  13365  13365
             13370  13370  13375  13380  13380  13415  13430
             13435  13465  13470  13475  13480  13500  13500
             13530  13555  13565  13565  13570  13570  13575
             13575  13580  13580  13585  13585  13615  13640
             13655  13655  13660  13660  14050  14155  14185
             14190  14195  15100  15155  17045
TO             585    640    730    735    790    865    985
              1155   1260   1395   1520   1655  12060  13210
             13240  13400  13435  13690  14045  14130  14165
             15125  15140  15150  17015
USING        15115  15155
VAL            615    620    830    835    840    845    850
               860
WRITE          645
```

WRITE/BAS Program Listing (TRSDOS Version)

```
1 '*********************************************************
2 '* WRITE/BAS - COMPUTER-ASSISTED INSTRUCTION SOFTWARE    *
3 '* ROBERT MASON, LT, SC, USN                             *
4 '* AIR FORCE INSTITUTE OF TECHNOLOGY                     *
5 '* SCHOOL OF SYSTEMS AND LOGISTICS                       *
6 '* MAY 1987                                              *
7 '* TANDY/RADIO SHACK MODEL IV VERSION 01.00.00           *
8 '*********************************************************
9 '
10 GOTO 10000   '   JUMP TO START OF MAIN PROGRAM
15 '
100 '**********************************************
105 '*              SUBROUTINES                   *
110 '*           (LINES 100-9999)                 *
115 '**********************************************
120 '
125 '   SUBROUTINE - PRINT SCREEN BOILERPLATE
130 CLS
          :PRINT@(0,0),QNAME$;
          :PRINT@(0,27),QTYPE$;
          :PRINT@(0 ,76),QP AGE$;
          :PRINT@(1,0),DASH$;
135 PRINT@(22,0),DASH$;
          :PRINT@(23,0),QINST$;
          :PRINT@(2,0),;
140 RETURN
145 '
150 '   SUBROUTINE - WAIT FOR LEGAL LETTER INPUT
155 A$=INPUT$(1)
160 IF A$=CHR$(5) THEN END
165 IF INSTR(LEGAL$,A$)=0 THEN 155
170 TEST%=ASC(A$)
175 IF TEST%>=97 AND TEST%<=122 THEN A$=CHR$(TEST%-32)
180 RETURN
185 '
190 '
195 '   SUBROUTINE - WAIT FOR <ENTER> INPUT
200 A$=INPUT$(1)
205 IF A$=CHR$(5) THEN END
210 IF A$<>CHR$(13) THEN 195
215 RETURN
220 '
225 '   SUBROUTINE - CHECK FOR LEGAL JUMP PAGE NUMBERS
230 IF PTEST%<1 OR PTEST%>9999 THEN FLAG%=1 ELSE FLAG%=0
235 RETURN
240 '
245 '   SUBROUTINE - FULL SCREEN INPUT ROUTINE
250 LC%=1
          :CC%=1
255 PRINT@(LC%+1,CC%-1),;
```

```
260 A$=INKEY$    'A$=INPUT$(1)
265 IF A$="" THEN 260
270 IF A$<>CHR$(13) THEN 305    '    <ENTER>
275 TEXT$(LC%)=LEFT$(TEXT$(LC%),CC%-1)
280 CC%=1
285 LC%=LC%+1
290 IF LC%>=21 THEN 540
295 PRINT CHR$(30);
          :GOTO 255
300 '
305 IF A$<>CHR$(8) THEN 340    '    BACKSPACE (<--)
310 IF CC%=1 THEN 255
315 TEXT$(LC%)=LEFT$(TEXT$(LC%),CC%-2)
320 PRINT A$;
325 CC%=CC%-1
330 GOTO 255
335 '
340 IF A$<>CHR$(9) AND A$<>CHR$(25) THEN 375    '    TAB (-->)
345 IF CC%>75 THEN 255    '    NOT ENOUGH SPACE TO TAB
350 TEXT$(LC%)=TEXT$(LC%)+STRING$(5,32)
355 CC%=CC%+5
360 PRINT STRING$(5,32);
365 GOTO 255
370 '
375 IF ASC(A$)<>5 THEN 395    '    CTRL<E> (END PAGE INPUT)
380 TEXT$(LC%)=LEFT$(TEXT$(LC%),CC%-1)
385 RETURN
390 '
395 IF ASC(A$)<>17 THEN 415    '    CTRL<Q> (QUIT PAGE EDITOR
    - NO SAVE)
400 ERASE TEXT$
405 RETURN
410 '
415 IF ASC(A$)<32 OR ASC(A$)>126 THEN 255    '    VALID ASCII
    CHARACTER
420 PRINT A$;
425 TEXT$(LC%)=TEXT$(LC%)+A$
430 CC%=CC%+1
435 IF CC%=81 THEN 450
440 GOTO 255
445 '
450 IF LC%=20 THEN 540
455 A$=INKEY$
          :IF A$="" THEN 455
460 IF A$<>CHR$(32) THEN 490    '    NO WORD WRAP REQUIRED
465 TEXT$(LC%)=LEFT$(TEXT$(LC%),80)
470 LC%=LC%+1
          :CC%=1
475 IF LC%=21 THEN 540
480 GOTO 255
485 '
490 TEST%=80
495 TEST$=MID$(TEXT$(LC%),TEST%,1)
```

```
500 IF INSTR(" /-",TEST$)=0 THEN TEST%=TEST%-1
         :GOTO 495 ELSE
505 TEXT$(LC%+1)=RIGHT$(TEXT$(LC%),80-TEST%)+A$
510 TEXT$(LC%)=LEFT$(TEXT$(LC%),TEST%)
515 IF TEST%<80 THEN PRINT@(LC%+1,TEST%),CHR$(30);
520 PRINT@(LC%+2,0),TEXT$(LC%+1);
525 CC%=82-TEST%
530 LC%=LC%+1
535 GOTO 255
540 PRINT@(23,0),CHR$(30);"          Page full!   Press Ctrl
    <E> to continue.";
         :PRINT@(23,0),;
545 A$=INKEY$
         :IF A$="" THEN 545
550 IF ASC(A$)<>5 THEN 545 ELSE LC%=20
         :RETURN
555 '
560 '   SUBROUTINE - CREATE LESSON TABLE FILE
565 OPEN "D",1,TEXT$,81
570 FIELD 1,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5 AS
     BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,46 AS
    DUMMY$
575 COUNT=0
580 LASTREC=LOF(1)
585 FOR I=1 TO LASTREC
590 GET 1,I
595 IF BUF1$=" " OR BUF1$="*" THEN 625
600 COUNT=COUNT+1
605 TYPE$(COUNT)=BUF1$
610 START%(COUNT)=LOC(1)
615 MORE%(COUNT)=VAL(BUF9$)
620 PAGE%(COUNT)=VAL(BUF2$)
625 NEXT I
630 CLOSE 1
635 OPEN "O",2,TABLE$
640 FOR I=1 TO COUNT
645 WRITE#2,PAGE%(I),START%(I),MORE%(I),TYPE$(I)
650 NEXT I
655 CLOSE 2
660 RETURN
665 '
670 '   SUBROUTINE - READ LESSON TABLE INTO MEMORY
675 QNAME$=TNAME$:QINST$=STRING$(20,32)+"Loading lesson tabl
    e...please wait."
         :QPAGE$=""
         :QTYPE$=""
         :GOSUB 125
680 PRINT@(23,0),;
685 OPEN "I",2,TABLE$
690 I=1
695 IF EOF(2) THEN 715
700 INPUT#2,PAGE%(I),START%(I),MORE%(I),TYPE$(I)
705 I=I+1
```

```
710 GOTO 695
715 CLOSE 2
720 NUMPAGES%=I-1
725 '   SORT TABLE INTO PAGE # SEQUENCE
730 FOR I=1 TO NUMPAGES%-1
735 FOR J=I+1 TO NUMPAGES%
740 IF PAGE%(I)>PAGE%(J) THEN ELSE 765
745 SWAP PAGE%(I),PAGE%(J)
750 SWAP START%(I),START%(J)
755 SWAP MORE%(I),MORE%(J)
760 SWAP TYPE$(I),TYPE$(J)
765 NEXT J
770 NEXT I
775 RETURN
780 '
785 '   SUBROUTINE - FIND CORRESPONDING TABLE SUBSCRIPT FOR
        QPAGE%
790 FOR I=1 TO NUMPAGES%
795 IF PAGE%(I)=QPAGE% THEN QMAT%=I
          :RETURN
800 NEXT I
805 QMAT%=-1
          :RETURN
810 '
815 '   SUBROUTINE - READ LESSON PAGE INTO MEMORY
820 GET 1,START%(QMAT%)
825 TYPE$=BUF1$
830 AJUMP%=VAL(BUF3$)
835 BJUMP%=VAL(BUF4$)
840 CJUMP%=VAL(BUF5$)
845 DJUMP%=VAL(BUF6$)
850 EJUMP%=VAL(BUF7$)
855 ANSWER$=BUF8$
860 MORE%=VAL(BUF9$)
865 FOR I=1 TO MORE%
870 GET 1,(START%(QMAT%)+I)
875 DUMMY$=BUF10$
880 TEXT$(I)=BUF11$
885 NEXT I
890 RETURN
895 '
900 '   SUBROUTINE - INPUT LESSON SCREEN
905 '   GET PAGE NUMBER
910 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=QDATA$
          :GOSUB 125
915 PRINT@(3,15),CHR$(30);
920 INPUT"Enter page number: ",PAGE:IF PAGE>9999 THEN 915 EL
    SE PAGE%=PAGE
925 IF PAGE%=9999 OR PAGE%=0 THEN RETURN
930 '   IF FIRST SCREEN, ENSURE NUMBERED AS PAGE 1
```

```
935 IF REC=1 AND PAGE%=1 OR REC>1 THEN 955
940 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
945 PRINT@(3,15),"The first page of a lesson must be page 1!
    ";
950 PRINT@(23,0),;
          :GOSUB 195
          :GOTO 905
955 '   ENSURE PAGE NUMBER IN LEGAL LIMITS
960 IF PAGE%>=1 AND PAGE%<=9999 THEN 980
965 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
970 PRINT@(3,15),PAGE%;" is not a valid page number!";
975 PRINT@(23,0),;
          :GOSUB 195
          :GOTO 905
980 '   ENSURE DUPLICATE PAGE NUMBER NOT BEING ENTERED
985 FOR I=1 TO NUMPAGES%
990 IF PAGE%=PAGE%(I) THEN 1010
995 NEXT I
1000 NUMPAGES%=NUMPAGES%+1
1005 GOTO 1030
1010 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
1015 PRINT@(3,5),"Page #";PAGE%;" has already been used! Do
     not duplicate page numbers!";
1020 PRINT@(23,0),;
          :GOSUB 195
          :GOTO 905
1025 '
1030 '   GET PAGE TYPE TO ENTER
1035 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=STR$(PAGE%)
          :QINST$=PROMPT$
          :GO SUB 125
1040 PRINT@(3,15),"<A> Input text page";
1045 PRINT@(4,15),"<B> Input multiple choice question page";
1050 PRINT@(5,15),"<C> Input true/false question page";
1055 PRINT@(23,0),;
          :LEGAL$="ABCabc"
          :GOSUB 150
1060 IF A$="A" THEN TYPE$="#"
          :QTYPE$=TP$
```

```
1065 IF A$="B" THEN TYPE$="?"
        :QTYPE$=MCQP$
1070 IF A$="C" THEN TYPE$="&"
        :QTYPE$=TFQP$
1075 QNAME$=TNAME$
        :QPAGE$=STR$(PAGE%)
        :QINST$=QEDIT$
        :GOSUB 125
1080 GOSUB 245
1085 '
1090 '   GET ADDITIONAL DATA REQUIRED FOR EACH PAGE TYPE
1095 PRINT@(23,0),CHR$(30);QDATA$;
1100 IF TYPE$="#" THEN 1115
1105 IF TYPE$="?" THEN 1150
1110 IF TYPE$="&" THEN 1255
1115 '   GET ADDITIONAL DATA FOR TEXT PAGE
1120 PRINT@(20,0),CHR$(30);
1125 PRINT@(21,0),CHR$(30);
1130 PRINT@(20,20),CHR$(30);
1135 INPUT"Jump-to page: ",AJUMP:IF AJUMP>9999 THEN 1130 ELS
     E AJUMP%=AJUMP
        :PTEST%=AJUMP%
        :GOSUB 225
1140 IF FLAG%=1 THEN 1120
1145 GOTO 1315
1150 '   GET ADDITIONAL DATA FOR MULTIPLE CHOICE QUESTION
         PAGE
1155 FOR I=15 TO 21
        :PRINT@(I,0),CHR$(30);
        :NEXT I
1160 PRINT@(15,20),"Correct answer: ";
        :LEGAL$="ABCDEabcde"
        :GOSUB 150
        :PRINT A$;
        :ANSWER$=A$
1165 PRINT@(16,20),CHR$(30);
1170 INPUT"A-jump page   : ",AJUMP
        :IF AJUMP>9999 THEN 1165 ELSE AJUMP%=AJUMP
        :PTEST%=AJUMP%
        :GOSUB 225
1175 IF FLAG%=1 THEN 1165
1180 PRINT@(17,20),CHR$(30);
1185 INPUT"B-jump page   : ",BJUMP
        :IF BJUMP>9999 THEN 1180 ELSE BJUMP%=BJUMP
        :PTEST%=BJUMP%
        :GOSUB 225
1190 IF FLAG%=1 THEN 1180
1195 PRINT@(18,20),CHR$(30);
1200 INPUT"C-jump page   : ",CJUMP
        :IF CJUMP>9999 THEN 1195 ELSE CJUMP%=CJUMP
        :PTEST%=CJUMP%
        :GOSUB 225
1205 IF FLAG%=1 THEN 1195
```

```
1210 PRINT@(19,20),CHR$(30);
1215 INPUT"D-jump page   : ",DJUMP
        :IF DJUMP>9999 THEN 1210 ELSE DJUMP%=DJUMP
        :PTEST%=DJUMP%
        :GOSUB 225
1220 IF FLAG%=1 THEN 1210
1225 PRINT@(20,20),CHR$(30);
1230 INPUT"E-jump page   : ",EJUMP
        :IF EJUMP>9999 THEN 1225 ELSE EJUMP%=EJUMP
        :PTEST%=EJUMP%
        :GOSUB 225
1235 IF FLAG%=1 THEN 1225
1240 TYPE$="?"
1245 GOTO 1315
1250 '
1255 '   GET ADDITIONAL DATA FOR TRUE/FALSE QUESTION PAGE
1260 FOR I=17 TO 21:PRINT@(I,0),CHR$(30);
        :NEXT I
1265 PRINT@(18,20),"Correct answer: ";:LEGAL$="TFtf"
        :GOSUB 150
        :PRINT A$;
        :ANSWER$=A$
1270 PRINT@(19,20),CHR$(30);
1275 INPUT"T-jump page   : ",AJUMP
        :IF AJUMP>9999 THEN 1270 ELSE AJUMP%=AJUMP
        :PTEST%=AJUMP%
        :GOSUB 225
1280 IF FLAG%=1 THEN 1270
1285 PRINT@(20,20),CHR$(30);
1290 INPUT"F-jump page   : ",BJUMP
        :IF BJUMP>9999 THEN 1285 ELSE BJUMP%=BJUMP
        :PTEST%=BJUMP%
        :GOSUB 225
1295 IF FLAG%=1 THEN 1285
1300 TYPE$="?"
1305 GOTO 1315
1310 '
1315 '   GET ALL OTHER PAGE DATA AND WRITE HEADER RECORD TO
         DISK
1320 MORE%=LC%
1325 LSET BUF1$=TYPE$
1330 LSET BUF2$=STR$(PAGE%)
1335 LSET BUF3$=STR$(AJUMP%)
1340 LSET BUF4$=STR$(BJUMP%)
1345 LSET BUF5$=STR$(CJUMP%)
1350 LSET BUF6$=STR$(DJUMP%)
1355 LSET BUF7$=STR$(EJUMP%)
1360 LSET BUF8$=ANSWER$
1365 LSET BUF9$=STR$(MORE%)
1370 LSET DUMMY$=""
1375 START%=REC
1380 PUT 1,REC
1385 '
```

```
1390 '   WRITE TEXT RECORDS TO DISK
1395 FOR I=1 TO MORE%
1400 REC=REC+1
1405 LSET BUF10$=" "
1410 LSET BUF11$=TEXT$(I)
1415 PUT 1,REC
1420 NEXT I
1425 REC=REC+1
1430 '
1435 '   UPDATE LESSON TABLE
1440 PAGE%(NUMPAGES%)=PAGE%
          :START%(NUMPAGES%)=START%
          :MORE%(NUMPAG ES%)=MORE%
          :TYPE$(NUMPAGES%)=TYPE$
1445 '   RETURN TO GET NEXT PAGE
1450 ERASE TEXT$
          :DIM TEXT$(20)
1455 GOTO 905
1460 '
1465 '   SUBROUTINE - GET DISK FILENAME AND PRINT ERRORS
1470 '   PRINT DISK INSTRUCTIONS
1475 QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
1480 PRINT@(3,10),"Insert the disk containing the lesson fil
     es in drive 1";
1485 PRINT@(23,0),;
1490 GOSUB 195
1495 '
1500 QTYPE$=""
          :QPAGE$=""
          :QINST$=QDATA$
          :GOSUB 125
1505 PRINT@(3,5),"Enter lesson name (maximum of 8 characters
      ; do not include";
          :PRINT@(4,5),;
1510 INPUT"extension; <ENTER> to abort): ",TNAME$
1515 IF TNAME$="" THEN CLOSE
          :ON ERROR GOTO 17000
          :GOTO 11000
1520 FOR I=1 TO LEN(TNAME$)
1525 TEST%=ASC(MID$(TNAME$,I,1))
1530 IF TEST%>=97 AND TEST%<=122 THEN MID$(TNAME$,I,1)=CHR$(
     TEST%-32)
1535 NEXT I
1540 '
1545 '   CHECK FOR NEEDED FILES ON DISK
1550 TEXT$=TNAME$+"/TXT"
1555 TABLE$=TNAME$+"/TAB"
1560 STUFILE$=TNAME$+"/STU"
1565 ON ERROR GOTO 1590
1570 IF INSTR(NEED$,"A")<>0 THEN QTEST$=TEXT$
```

```
          :OPEN "I",2,QTEST$
          :CLOSE 2
1575 IF INSTR(NEED$,"B")<>0 THEN QTEST$=TABLE$
          :OPEN "I",2,QTEST$
          :CLOSE 2
1580 IF INSTR(NEED$,"C")<>0 THEN QTEST$=STUFILE$
          :OPEN "I",2,QTEST$:CLOSE 2
1585 CLOSE
          :ON ERROR GOTO 17000
          :RETURN
1590 '
1595 QNAME$="ERROR"
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
1600 IF ERR=53 THEN PRINT@(3,15),QTEST$;" is not on this dis
     k!";
          :GOTO 1620
1605 IF ERR=57 THEN PRINT@(3,15),"A device input/output erro
     r has occurred!";
          :GOTO 1620
1610 IF ERR=64 THEN PRINT@(3,15),TNAME$;" is not a valid
     lesson name!";
          :GOTO 1620
1615 PRINT@(3,15),"An unknown disk error has occurred!";
1620 RESUME 1625
1625 PRINT@(23,0),;
1630 GOSUB 195
1635 GOTO 1500
1640 CLOSE
          :ON ERROR GOTO 17000
          :RETURN
1645 '
1650 '   SUBROUTINE - DISPLAY LESSON PAGE
1655 FOR I=1 TO MORE%(QMAT%)
1660 PRINT TEXT$(I);
1665 NEXT I
1670 RETURN
1675 '
10000 '*********************************************
10005 '*   CONSTANT TABLE AND DEFINED FUNCTIONS     *
10010 '*         (LINES 10000-10999)                *
10015 '*********************************************
10020 '
10025 OPTION BASE 1
          :ON ERROR GOTO 17000
10030 DASH$=STRING$(80,45)
10035 PROMPT$=STRING$(22,32)+"Press <letter> of your choice.
      "
10040 QDATA$=STRING$(14,32)+"Enter requested data and press
      <ENTER> to continue."
10045 QEDIT$=STRING$(10,32)+"Enter Text"+STRING$(10,32)+"Ctr
      l<E> to End"+STRING$(10,32)+"Ctrl<Q> to Quit"
```

```
10050 MCQP$="Multiple Choice Question Page"
10055 TP$=STRING$(10,32)+"Text Page"
10060 TFQP$=STRING$(2,32)+"True/False Question Page"
10065 ENTER$=STRING$(22,32)+"Press <ENTER> to continue."
10070 YN$=STRING$(10,32)+"<Y>es     <N>o"+STRING$(25,32)
10075 '
11000 '**********************************************
11005 '*            MAIN PROGRAM                    *
11010 '*         (LINES 11000-39999)                *
11015 '**********************************************
11020 '
11025 '   INITIALIZE COUNTERS
11030 NUMPAGES%=0
11035 '
11040 '   PRINT MAIN MENU AND ROUTE TO PROPER PORTION OF
          PROGRAM
11045 QNAME$="MAIN MENU"
          :QPAGE$=""
          :QINST$=PROMPT$
          :QTYPE$=""
          :GOSUB 125
11050 PRINT@(3,15),"<A>  Create a lesson file."
11055 PRINT@(4,15),"<B>  Edit a lesson file."
11060 PRINT@(5,15),"<C>  Print a lesson file."
11065 PRINT@(6,15),"<D>  Print a student file report."
11070 PRINT@(7,15),"<E>  Exit WRITE program."
11075 PRINT@(23,0),;
11080 LEGAL$="ABCDEabcde"
          :GOSUB 150
11085 IF A$="A" THEN 12000 ELSE IF A$="B" THEN 13000 ELSE IF
      A$="C" THEN 14000 ELSE IF A$="D" THEN 15000 ELSE IF A$
      ="E" THEN 16000
11090 '
12000 '   CREATE A LESSON FILE
12005 DIM TEXT$(20),PAGE%(200),MORE%(200),TYPE$(200),START%(
      200)
12010 '   PRINT DISK INSTRUCTIONS
12015 QNAME$="CREATE A LESSON FILE"
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
12020 PRINT@(3,15),"Insert a properly formatted disk in driv
      e 1."
12025 PRINT@(23,0),;
          :GOSUB 195
12030 '
12035 '   GET LESSON FILE NAME
12040 QNAME$="CREATE A LESSON FILE"
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=QDATA$
          :GOSUB 125
```

```
12045 PRINT@(3,5),"Enter lesson name (maximum of 8 character
      s; do not include";
          :PRINT@(4,5),;
12050 INPUT"extension; <ENTER> to abort): ",TNAME$
12055 IF TNAME$="" THEN GOTO 11000
12060 FOR I=1 TO LEN(TNAME$)
12065 TEST%=ASC(MID$(TNAME$,I,1))
12070 IF TEST%>=97 AND TEST%<=122 THEN MID$(TNAME$,I,1)=CHR$
      (TEST%-32)
12075 NEXT I
12080 TEXT$=TNAME$+"/TXT:1"
12085 TABLE$=TNAME$+"/TAB:1"
12090 '
12095 '   OPEN LESSON FILE BUFFER, INITIALIZE RECORD
          COUNTER, AND START
12100 '   PAGE INPUT
12105 ON ERROR GOTO 12180
12110 OPEN "D",1,TEXT$,81
12115 FIELD 1,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5
      AS BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,4
      6 AS DUMMY$
12120 FIELD 1,1 AS BUF10$,80 AS BUF11$
12125 ON ERROR GOTO 17000
12130 REC=1
12135 GOSUB 900
12140 CLOSE
12145 '
12150 '   CREATE LESSON TABLE FILE
12155 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=STRING$(14,32)+"Writing lesson files to di
           sk...please wait."
          :GOSUB 125
12160 GOSUB 560
12165 CLOSE
          :ERASE TEXT$,PAGE%,MORE%,TYPE$,START%
12170 CLS
          :GOTO 11025    '    RETURN TO MAIN MENU
12175 '
12180 '   ERROR HANDLING ROUTINE FOR FILE NAME TO CREATE
12185 QNAME$="ERROR"
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
12190 IF ERR=57 THEN PRINT@(3,15),"A device input/output err
      or has occurred!"
          :GOTO 12205
12195 IF ERR=64 THEN PRINT@(3,15),TNAME$;" is not a valid le
      sson name!";
          :GOTO 12205
12200 PRINT@(3,15),"An unknown disk error has occurred!";
```

```
12205 PRINT@(23,0),;
          :GOSUB 195
12210 RESUME 12215
12215 CLOSE
          :ON ERROR GOTO 17000
          :GOTO 12015
13000 '   EDIT A LESSON FILE
13005 QNAME$="EDIT A LESSON FILE"
13010 '   GET LESSON FILE NAME
13015 NEED$="AB"
          :GOSUB 1465
13020 DIM TEXT$(20),PAGE%(200),START%(200),MORE%(200),TYPE$(
      200)
13025 '
13030 '   READ TABLE FILE INTO MEMORY
13035 GOSUB 670
13040 CLOSE:
          OPEN "D",1,TEXT$,81
13045 FIELD 1,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5
      AS BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,4
      6 AS DUMMY$
13050 FIELD 1,1 AS BUF10$,80 AS BUF11$
13055 '
13060 '   DISPLAY EDIT MAIN MENU
13065 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=PROMPT$
          :GOSUB 125
13070 PRINT@(3,15),"<A>  Add screen to lesson."
13075 PRINT@(4,15),"<B>  Delete screen from lesson."
13080 PRINT@(5,15),"<C>  Modify existing screen."
13085 PRINT@(6,15),"<D>  Return to WRITE main menu."
13090 PRINT@(23,0),;
13095 LEGAL$="ABCDabcd"
          :GOSUB 150
13100 IF A$="A" THEN 13105 ELSE IF A$="B" THEN 13130 ELSE IF
      A$="C" THEN 13270 ELSE IF A$="D" THEN 13700
13105 '   ADD SCREEN TO LESSON
13110 '   GET CURRENT LAST RECORD
13115 REC=LOF(1)+1
13120 GOSUB 900
13125 GOTO 13060
13130 '   DELETE PAGE FROM EXISTING LESSON FILE
13135 '   GET PAGE NUMBER
13140 QNAME$=TNAME$
          :QTYPE$="DELETE A PAGE"
          :QPAGE$=""
          :QINST$=QDATA$
          :GOSUB 125
13145 PRINT@(3,15),CHR$(30);
13150 INPUT"Enter page number to delete: ",PAGE
          :IF PAGE>9999 THEN 13145 ELSE PAGE%=PAGE
```

```
13155 '   FIND TABLE SUBSCRIPT/READ LESSON PAGE INTO MEMORY
13160 QPAGE%=PAGE%
        :GOSUB 785
13165 IF QMAT%=-1 THEN PRINT@(5,15),"Page not found in lesso
      n table!";
        :PRINT@(23,0),CHR$(30);ENTER$;
        :PRINT@(23,0),;
        :GOSUB 195
        :GOTO 13060
13170 GOSUB 815
13175 '   DISPLAY PAGE
13180 QNAME$=TNAME$
        :QTYPE$="DELETE A PAGE"
        :QPAGE$=STR$(PAGE%)
        :QINST$=STRING$(15,32)+"<D> to Delete Page"+STRING
         $(15,32)+"<Q> to Abort"
        :GOSUB 125
13185 GOSUB 1650
13190 '   GET DELETE DECISION
13195 PRINT@(23,0),;
13200 LEGAL$="QqDd"
        :GOSUB 150
13205 IF A$="Q" THEN ERASE TEXT$
        :DIM TEXT$(20)
        :GOTO 13060
13210 FOR REC=START%(QMAT%) TO START%(QMAT%)+MORE%(QMAT%)
13215 LSET BUF10$="*"
13220 LSET BUF11$=STRING$(80,42)
13225 PUT 1,REC
13230 NEXT REC
13235 '   UPDATE LESSON TABLE
13240 FOR I=QMAT% TO NUMPAGES%
13245 PAGE%(I)=PAGE%(I+1)
        :START%(I)=START%(I+1)
        :MORE%(I)=MORE%(I+1)
        :TYPE$(I)=TYPE$(I+1)
        :NEXT I
13250 NUMPAGES%=NUMPAGES%-1
13255 ERASE TEXT$
        :DIM TEXT$(20)
13260 GOTO 13060
13265 '
13270 '   MODIFY EXISTING PAGE
13275 '   GET PAGE NUMBER
13280 QNAME$=TNAME$
        :QTYPE$="MODIFY A PAGE"
        :QPAGE$=""
        :QINST$=QDATA$
        :GOSUB 125
13285 PRINT@(3,15),CHR$(30);
13290 INPUT"Enter page number to modify: ",PAGE
        :IF PAGE>9999 THEN 13285 ELSE PAGE%=PAGE
13295 '   FIND TABLE SUBSCRIPT/READ LESSON PAGE INTO MEMORY
```

```
13300 QPAGE%=PAGE%
          :GOSUB 785
13305 IF QMAT%=-1 THEN PRINT@(5,15),"Page not found in lesso
      n table!";
          :PRINT@(23,0),CHR$(30);ENTER$;
          :PRINT@(23,0),;
          :GOSUB 195
          :GOTO 13060
13310 GOSUB 815
13315 '   DISPLAY PAGE
13320 QNAME$=TNAME$
          :QTYPE$="MODIFY A PAGE"
          :QPAGE$=STR$(PAGE%)
          :QINST$="   Ctrl <E>nd text edit   Ctrl <Q>uit and
           cancel   Ctrl <D>elete character"
          :GOSUB 125
13325 GOSUB 1650
13330 PRINT@(2,0),;
13335 A$=INKEY$
13340 IF A$="" THEN 13335
13345 IF ASC(A$)=17 THEN ERASE TEXT$
          :DIM TEXT$(20)
          :GOTO 13060   '   QUIT EDITOR
13350 IF ASC(A$)=5 THEN GOTO 13390   '   END TEXT EDITING
13355 IF ASC(A$)=11 THEN IF ROW(1)=2 THEN 13335 ELSE PRINT@(
      ROW(1)-1,POS(1)-1),;
          :GOTO 13335   '   UP ARROW
13360 IF ASC(A$)=10 THEN IF ROW(1)=MORE%+1 THEN 13335 ELSE P
      RINT@(ROW(1)+1,POS(1)-1),;
          :GOTO 13335   '   DOWN ARROW
13365 IF ASC(A$)=8 THEN IF POS(1)=1 THEN 13335 ELSE PRINT@(
      ROW(1),POS(1)-2),;
          :GOTO 13335   '   LEFT ARROW
13370 IF ASC(A$)=9 THEN IF POS(1)=80 THEN 13335 ELSE PRINT@
      (ROW(1),POS(1)),;
          :GOTO 13335   '   RIGHT ARROW
13375 IF ASC(A$)=4 THEN TPOS=POS(1)
          :TROW=ROW(1)
          :TEXT$(TROW-1)=LEFT$(TEXT$(TROW-1),TPOS-1)+RIGHT$(
           TEXT$(TROW-1),80-TPOS)+CHR$(32)
          :PRINT@(TROW,0),TEXT$(TROW-1);
          :PRINT@(TROW,TPOS-1),;
          :GOTO 13335   '   DELETE CHARACTER
13380 IF ASC(A$)>=32 AND ASC(A$)<=126 THEN MID$(TEXT$(ROW(1)
      -1),POS(1),1)=A$
          :PRINT A$;
          :IF ROW(1)=MORE%+2 AND POS(1)=1 THEN PRINT CHR$(24
           );
          :GOTO 13335   '   OVERSTRIKE WITH VALID ASCII
           CHARACTER
13385 GOTO 13335   '   INVALID KEY ENTRY
13390 '   END TEXT EDITING - CHANGE ADDITIONAL DATA?
13395 '   CHANGE PAGE NUMBER?
```

```
13400 FOR I=19 TO 21
          :PRINT@(I,0),CHR$(30),;
          :NEXT I
13405 PRINT@(20,20),CHR$(30);"Page number: ";PAGE%;
          :PRINT@(23,0),CHR$(30);" Change page number?";YN$;
          :PRINT@(23,0),;
13410 LEGAL$="YyNn"
          :GOSUB 150
13415 IF A$<>"Y" THEN 13460
13420 PRINT@(23,0),CHR$(30);QDATA$;
13425 PRINT@(20,20),CHR$(30);"New page number: ";
13430 INPUT"",PAGE
          :IF PAGE>9999 THEN 13425
13435 FOR I=1 TO NUMPAGES%
          :IF PAGE=PAGE%(I) THEN 13445 ELSE NEXT I
13440 PAGE%=PAGE
          :PAGE%(QMAT%)=PAGE%
          :GOTO 13405
13445 PRINT@(20,20),CHR$(30);PAGE;" already used!";
13450 PRINT@(23,0),CHR$(30);ENTER$;
          :PRINT@(23,0),;
          :GOSUB 195
13455 GOTO 13405
13460 '   ASSIGN TEMPORARY TYPE$ TO T/F QUESTION
13465 IF TYPE$="?" AND (ANSWER$="T" OR ANSWER$="F")  THEN TY
      PE$="&"
13470 IF TYPE$="#" THEN 13485
13475 IF TYPE$="?" THEN 13510
13480 IF TYPE$="&" THEN 13595
13485 '   TEXT PAGE - CHANGE JUMP PAGE?
13490 PRINT@(20,20),CHR$(30);"Jump page: ";AJUMP%;
13495 PRINT@(23,0),CHR$(30);" Change jump page?";YN$;
          :PRINT@(23,0),;
          :LEGAL$="YyNn"
          :GOSUB 150
13500 IF A$<>"Y" THEN 13670 ELSE PRINT@(23,0),CHR$(30);QDATA
      $;
          :PRINT@(20,20),CHR$(30);"Jump page: ";
          :INPUT"",AJUMP:IF AJUMP<10000 AND AJUMP>0 THEN AJU
           MP%=AJUMP ELSE GOTO 13500
13505 GOTO 13485
13510 '   MULTIPLE CHOICE QUESTION
13515 '   CHANGE CORRECT ANSWER?
13520 PRINT@(20,20),CHR$(30);"Correct answer: ";ANSWER$;
13525 PRINT@(23,0),CHR$(30);" Change correct answer?";YN$;
          :PRINT@(23,0),;
          :LEGAL$="YyNn"
          :GOSUB 150
13530 IF A$<>"Y" THEN 13540
```

```
13535 PRINT@(23,0),CHR$(30);PROMPT$;
          :PRINT@(20,20),CHR$(30);"Correct answer: ";
          :LEGAL$="ABCDEabcde"
          :GOSUB 150:PRINT A$;
          :ANSWER$=A$
          :GOTO 13520
13540 '   CHANGE ANSWER JUMP PAGES?
13545 PRINT@(20,0),"    AJUMP: ";AJUMP%;"    BJUMP: ";BJUMP%
      ;"    CJUMP: ";CJUMP%;"    DJUMP: ";DJUMP%;"    EJUMP:
      ";EJUMP%;
13550 PRINT@(23,0)," Change answer jump pages?";YN$;
          :PRINT@(23,0),;
          :LEGAL$="YyNn"
          :GOSUB 150
13555 IF A$<>"Y" THEN 13670
13560 PRINT@(23,0),CHR$(30);PROMPT$;
          :PRINT@(23,0)," Which one?";
          :PRINT@(23,0),;
          :LEGAL$="ABCDEabcde"
          :GOSUB 150
          :PRINT@(24,1),CHR$(30);QDATA$;
          :PRINT@(20,0),CHR$(30);
13565 IF A$="A" THEN PRINT@(20,20),CHR$(30);"AJUMP: ";
          :INPUT"",AJUMP
          :IF AJUMP>0 AND AJUMP<10000 THEN AJUMP%=AJUMP ELSE
           GOTO 13565
13570 IF A$="B" THEN PRINT@(20,20),CHR$(30);"BJUMP: ";
          :INPUT"",BJUMP
          :IF BJUMP>0 AND BJUMP<10000 THEN BJUMP%=BJUMP ELSE
           GOTO 13570
13575 IF A$="C" THEN PRINT@(20,20),CHR$(30);"CJUMP: ";
          :INPUT"",CJUMP
          :IF CJUMP>0 AND CJUMP<10000 THEN CJUMP%=CJUMP ELSE
           GOTO 13575
13580 IF A$="D" THEN PRINT@(20,20),CHR$(30);"DJUMP: ";
          :INPUT"",DJUMP
          :IF DJUMP>0 AND DJUMP<10000 THEN DJUMP%=DJUMP ELSE
           GOTO 13580
13585 IF A$="E" THEN PRINT@(20,20),CHR$(30);"EJUMP: ";
          :INPUT"",EJUMP
          :IF EJUMP>0 AND EJUMP<10000 THEN EJUMP%=EJUMP ELSE
           GOTO 13585
13590 GOTO 13540
13595 '   TRUE/FALSE QUESTION
13600 '   CHANGE CORRECT ANSWER?
13605 PRINT@(20,20),CHR$(30);"Correct answer: ";ANSWER$;
13610 PRINT@(23,0),CHR$(30);" Change correct answer?";YN$;
          :PRINT@(23,0),;
          :LEGAL$="YyNn"
          :GOSUB 150
13615 IF A$<>"Y" THEN 13625
```

```
13620 PRINT@(23,0),CHR$(30);PROMPT$;
        :PRINT@(20,0),CHR$(30);
        :PRINT@ (20,20),"Correct answer: ";
        :LEGAL$="TtFf"
        :GOSUB 150:PRINT A$;
        :ANSWER$=A$
        :GOTO 13600
13625 '   CHANGE ANSWER JUMP PAGES?
13630 PRINT@(20,0),CHR$(30);STRING$(24,32);"TJUMP: "; AJUMP%
      ; "              FJUMP: ";BJUMP%;
13635 PRINT@(23,0)," Change answer jump pages?";YN$;
        :PRINT@(23,0),;
        :LEGAL$="YyNn"
        :GOSUB 150
13640 IF A$<>"Y" THEN TYPE$="?"
        :GOTO 13670
13645 PRINT@(23,0),CHR$(30);PROMPT$;
        :PRINT@(23,0)," Which one?";
        :PRINT@(23,0),;:LEGAL$="TtFf"
        :GOSUB 150
13650 PRINT@(20,0),CHR$(30);
13655 IF A$="T" THEN PRINT@(20,20),CHR$(30);"TJUMP: ";
        :INPUT"",AJUMP
        :IF AJUMP>0 AND AJUMP<10000 THEN AJUMP%=AJUMP ELSE
         GOTO 13655
13660 IF A$="F" THEN PRINT@(20,20),CHR$(30);"FJUMP: ";
        :INPUT"",BJUMP
        :IF BJUMP>0 AND BJUMP<10000 THEN BJUMP%=BJUMP ELSE
         GOTO 13660
13665 GOTO 13630
13670 '   END EDIT - WRITE PAGE TO DISK
13675 LSET BUF1$=TYPE$
        :LSET BUF2$=STR$(PAGE%)
        :LSET BUF3$=STR$(AJUMP%)
        :LSET BUF4$=STR$(BJUMP%)
        :LSET BUF5$=STR$(CJUMP%)
        :LSET BUF6$=STR$(DJUMP%)
        :LSET BUF7$=STR$(EJUMP%)
        :LSET BUF8$=ANSWER$
        :LSET BUF9$=STR$(MORE%)
        :LSET DUMMY$=""
13680 REC=START%(QMAT%)
13685 PUT 1,REC
13690 FOR I=1 TO MORE%
        :REC=REC+1
        :LSET BUF10$=" "
        :LSET BUF11$=TEXT$(I)
        :PUT 1,REC
        :NEXT I
13695 ERASE TEXT$
        :DIM TEXT$(20)
        :GOTO 13060
13700 '  EXIT EDITOR - RETURN TO MAIN MENU
```

```
13705 QNAME$="EXIT EDITOR"
          :QPAGE$=""
          :QTYPE$=""
          :QINST$=STRING$(14,32)+"Writing lesson files to di
           sk...please wait."
          :GOSUB 125
13710 CLOSE
13715 GOSUB 560
13720 CLOSE
13725 ERASE PAGE%,START%,MORE%,TYPE$,TEXT$
13730 GOTO 11025
13735 '
14000 '   PRINT A LESSON FILE
14005 DIM TEXT$(20),PAGE%(200),MORE%(200),TYPE$(200),START%(
      200)
14010 '   GET LESSON FILE NAME
14015 QNAME$="PRINT A LESSON FILE"
14020 NEED$="AB"
14025 GOSUB 1465
14030 GOSUB 670
14035 '   ASSIGN QUESTION NUMBERS
14040 QNUM%=1
14045 FOR I=1 TO NUMPAGES%
14050 IF TYPE$(I)<>"?" THEN 14065
14055 QNUM%(I)=QNUM%
14060 QNUM%=QNUM%+1
14065 NEXT I
14070 '
14075 QNAME$=TNAME$
          :QTYPE$="PRINT A LESSON FILE"
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
14080 PRINT@(3,15),"Ensure printer is ready for printing.";
14085 PRINT@(23,0),;
14090 GOSUB 195
14095 '
14100 '   OPEN TEXT FILE
14105 OPEN "D",1,TEXT$,81
14110 FIELD 1,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5
      AS BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,4
      6 AS DUMMY$
14115 FIELD 1,1 AS BUF10$,80 AS BUF11$
14120 '
14125 '   READ PAGES OF TEXT AND OUTPUT TO PRINTER
14130 FOR QMAT%=1 TO NUMPAGES%
14135 GOSUB 815
14140 LPRINT DASH$
14145 LPRINT
14150 LPRINT "Page # ";PAGE%(QMAT%);
14155 IF TYPE$(QMAT%)="?" THEN LPRINT "/ Question #";QNUM%(Q
      MAT%) ELSE LPRINT
14160 LPRINT
```

```
14165 FOR J=1 TO MORE%
14170 LPRINT TEXT$(J)
14175 NEXT J
14180 LPRINT
14185 IF TYPE$(QMAT%)="#" THEN LPRINT "Jump:";AJUMP%
          :GOTO 14200
14190 IF TYPE$(QMAT%)="?" AND INSTR("TF",ANSWER$)<>0 THEN LP
      RINT"T-jump:";AJUMP%;"  F-jump:";BJUMP%; TAB(69);"Corr
      ect: ";ANSWER$
          :GOTO 14200
14195 IF TYPE$(QMAT%)="?" AND INSTR("ABCDE",ANSWER$)<>0 THEN
       LPRINT"A-jump:";AJUMP%;"  B-jump:";BJUMP%;"  C-jump:
      ";CJUMP%;"  D-jump:";DJUMP%;"  E-jump:";EJUMP%;TAB(69)
      ;"Correct: ";ANSWER$
14200 LPRINT
14205 ERASE TEXT$
          :DIM TEXT$(20)
14210 NEXT QMAT%
14215 LPRINT DASH$
14220 LPRINT CHR$(12);
14225 CLOSE
14230 ERASE PAGE%,START%,MORE%,TYPE$,TEXT$
14235 CLS
          :GOTO 11025    '    RETURN TO MAIN MENU
14240 '
15000 '   PRINT A STUDENT FILE REPORT
15005 QNAME$="PRINT A STUDENT FILE REPORT"
15010 NEED$="C"
          :GOSUB 1465
15015 '
15020 QNAME$=TNAME$
          :QTYPE$="PRINT A STUDENT FILE"
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
15025 PRINT@(3,15),"Ensure printer is ready for printing.";
15030 PRINT@(23,0),;
15035 GOSUB 195
15040 '
15045 '   PRINT HEADER INFORMATION
15050 LPRINT STRING$(26,32);TNAME$;" STUDENT FILE REPORT";TA
      B(65);DATE$
15055 LPRINT
15060 LPRINT DASH$
15065 LPRINT"STUDENT NAME/DATE   ASKED   RIGHT   WRONG    PCT
           WRONG QUESTIONS-RESPONSES"
15070 LPRINT DASH$
15075 LPRINT
15080 '
15085 '   PRINT STUDENT INFORMATION
15090 OPEN "I",1,STUFILE$
15095 DIM WQUEST%(100),WQUEST$(100)
15100 IF EOF(1) THEN 15185
```

```
15105 INPUT#1,STUDENT$,QDATE$,TCOUNT%,RCOUNT%,WCOUNT%
15110 PCT=(RCOUNT%/TCOUNT%)*100
15115 LPRINT USING "\       \ \     \    ###    ###    ###
      ###.#";STUDENT$,QDATE$,TCOUNT%,RCOUNT%,WCOUNT%,PCT;
15120 IF WCOUNT%=0 GOTO 15170
15125 FOR I=1 TO WCOUNT%
15130 INPUT#1,WQUEST%(I),WQUEST$(I)
15135 NEXT I
15140 FOR I=1 TO WCOUNT% STEP 5
15145 LPRINT TAB(51);
15150 FOR J=0 TO 4
15155 IF WQUEST%(I+J)<>0 THEN LPRINT USING "###_-! ";WQUEST%
      (I+J);WQUEST$(I+J);
15160 NEXT J
15165 NEXT I
15170 LPRINT
          :LPRINT
15175 ERASE WQUEST%,WQUEST$
15180 GOTO 15095
15185 '   END OF REPORT ROUTINE
15190 LPRINT DASH$
15195 LPRINT CHR$(12);
15200 ERASE WQUEST%,WQUEST$
15205 CLOSE
          :CLS
          :GOTO 11025   '   RETURN TO MAIN MENU
15210 '
16000 '   EXIT PROGRAM ROUTINE
16005 '
16010 CLS
          :END
17000 '   PROGRAM FATAL ERROR ROUTINE
17005 STAR$=STRING$(10,32)+STRING$(60,42)
17010 CLS
17015 PRINT@(3,0),STAR$
          :FOR I=4 TO 15
          :PRINT@(I,10),"**";
          :PRINT@(I,68),"**";
          :NEXT I
          :PRINT@(15,0),STAR$
17020 PRINT@(5,21),"FATAL PROGRAM ERROR DURING EXECUTION";
17025 PRINT@(7,25),"Error code ";ERR;" in line ";ERL;
17030 PRINT@(10,15),"Retain above data and refer to WRITE/BA
      S User's Guide";
17035 PRINT@(13,23),"Press <ENTER> to restart program.";
17040 PRINT@(13,22),;
17045 A$=INKEY$
          :IF A$<>CHR$(13) THEN 17045
17050 ERASE TEXT$,PAGE%,MORE%,TYPE$,START%
17055 CLOSE
          :RESUME 11000
17060 END
```

WRITE/BAS Program Listing (MSDOS Version)

```
1 '**********************************************************
2 '* WRITE/BAS - COMPUTER-ASSISTED INSTRUCTION SOFTWARE      *
3 '* ROBERT MASON, LT, SC, USN                               *
4 '* AIR FORCE INSTITUTE OF TECHNOLOGY                       *
5 '* SCHOOL OF SYSTEMS AND LOGISTICS                         *
6 '* MAY 1987                                                *
7 '* IBM/PC VERSION 01.00.00.                                *
8 '**********************************************************
9 '
10 GOTO 10000    '   JUMP TO START OF MAIN PROGRAM
15 '
100 '*******************************************
105 '*              SUBROUTINES                *
110 '*           (LINES 100-9999)              *
115 '*******************************************
120 '
125 '   SUBROUTINE - PRINT SCREEN BOILERPLATE
130 CLS
          :LOCATE 1,1
          :PRINT QNAME$;
          :LOCATE 1,28
          :PRINT QTYPE$;
          :LOCATE 1,77
          :PRINT QPAGE$;
          :LOCATE 2,1
          :PRINT DASH$;
135 LOCATE 23,1
          :PRINT DASH$;
          :LOCATE 24,1
          :PRINT QINST$;
          :LOCATE 3,1
140 RETURN
145 '
150 '   SUBROUTINE - WAIT FOR LEGAL LETTER INPUT
155 A$=INPUT$(1)
160 IF A$=CHR$(5) THEN END
165 IF INSTR(LEGAL$,A$)=0 THEN 155
170 TEST%=ASC(A$)
175 IF TEST%>=97 AND TEST%<=122 THEN A$=CHR$(TEST%-32)
180 RETURN
185 '
190 '
195 '   SUBROUTINE - WAIT FOR <ENTER> INPUT
200 A$=INPUT$(1)
205 IF A$=CHR$(5) THEN END
210 IF A$<>CHR$(13) THEN 195
215 RETURN
220 '
225 '   SUBROUTINE - CHECK FOR LEGAL JUMP PAGE NUMBERS
230 IF PTEST%<1 OR PTEST%>9999 THEN FLAG%=1 ELSE FLAG%=0
```

```
235 RETURN
240 '
245 '   SUBROUTINE - FULL SCREEN INPUT ROUTINE
250 LC%=1
          :CC%=1
255 LOCATE LC%+2,CC%
260 A$=INKEY$    'A$=INPUT$(1)
265 IF A$="" THEN 260
270 IF A$<>CHR$(13) THEN 305    '   <ENTER>
275 TEXT$(LC%)=LEFT$(TEXT$(LC%),CC%-1)
280 CC%=1
285 LC%=LC%+1
290 IF LC%>=21 THEN 540
295 PRINT CHR$(30);
          :GOTO 255
300 '
305 IF A$<>CHR$(8) AND A$<>CHR$(0)+CHR$(75) AND A$<>CHR$(0)+
    CHR$(83) THEN 340    '   BACKSPACE (<--)
310 IF CC%=1 THEN 255
315 TEXT$(LC%)=LEFT$(TEXT$(LC%),CC%-2)
320 PRINT CHR$(29);CHR$(32);CHR$(29);
325 CC%=CC%-1
330 GOTO 255
335 '
340 IF A$<>CHR$(9) AND A$<>CHR$(0)+CHR$(77) THEN 375
    '   TAB (-->)
345 IF CC%>75 THEN 255    '   NOT ENOUGH SPACE TO TAB
350 TEXT$(LC%)=TEXT$(LC%)+STRING$(5,32)
355 CC%=CC%+5
360 PRINT STRING$(5,32);
365 GOTO 255
370 '
375 IF ASC(A$)<>5 THEN 395    '   CTRL<E> (END PAGE INPUT)
380 TEXT$(LC%)=LEFT$(TEXT$(LC%),CC%-1)
385 RETURN
390 '
395 IF ASC(A$)<>17 THEN 415    '   CTRL<Q> (QUIT PAGE EDITOR
    - NO SAVE)
400 ERASE TEXT$
405 RETURN
410 '
415 IF ASC(A$)<32 OR ASC(A$)>126 THEN 255    '   VALID ASCII
    CHARACTER
420 PRINT A$;
425 TEXT$(LC%)=TEXT$(LC%)+A$
430 CC%=CC%+1
435 IF CC%=81 THEN 450
440 GOTO 255
445 '
450 IF LC%=20 THEN 540
455 A$=INKEY$
          :IF A$="" THEN 455
```

```
460 IF A$=CHR$(8) OR A$=CHR$(0)+CHR$(75) OR A$=CHR$(0)+CHR$(
    83) THEN 305 ELSE IF A$<>CHR$(32) THEN 490    '    NO WORD
    WRAP REQUIRED
465 TEXT$(LC%)=LEFT$(TEXT$(LC%),80)
470 LC%=LC%+1
          :CC%=1
475 IF LC%=21 THEN 540
480 GOTO 255
485 '
490 TEST%=80
495 TEST$=MID$(TEXT$(LC%),TEST%,1)
500 IF INSTR(" /-",TEST$)=0 THEN TEST%=TEST%-1
          :GOTO 495 ELSE
505 TEXT$(LC%+1)=RIGHT$(TEXT$(LC%),80-TEST%)+A$
510 TEXT$(LC%)=LEFT$(TEXT$(LC%),TEST%)
515 LOCATE LC%+2,TEST%+1
          :PRINT STRING$(81-POS(0),32);
520 LOCATE LC%+3,1
          :PRINT TEXT$(LC%+1);
525 CC%=82-TEST%
530 LC%=LC%+1
535 GOTO 255
540 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT"          Page full!   Press Ctrl<E> to con
           tinue.";
          :LOCATE 24,1
545 A$=INKEY$
          :IF A$="" THEN 545
550 IF ASC(A$)<>5 THEN 545 ELSE LC%=20
          :RETURN
555 '
560 '   SUBROUTINE - CREATE LESSON TABLE FILE
565 OPEN TEXT$ AS #1 LEN=81
570 FIELD #1,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5 A
    S BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,46 A
    S DUMMY$
575 COUNT=0
580 LASTREC=LOF(1)/81
585 FOR I=1 TO LASTREC
590 GET 1,I
595 IF BUF1$=" " OR BUF1$="*" THEN 625
600 COUNT=COUNT+1
605 TYPE$(COUNT)=BUF1$
610 START%(COUNT)=LOC(1)
615 MORE%(COUNT)=VAL(BUF9$)
620 PAGE%(COUNT)=VAL(BUF2$)
625 NEXT I
630 CLOSE 1
635 OPEN TABLE$ FOR OUTPUT AS #1
640 FOR I=1 TO COUNT
645 WRITE#1,PAGE%(I),START%(I),MORE%(I),TYPE$(I)
```

```
650 NEXT I
655 CLOSE 2
660 RETURN
665 '
670 '   SUBROUTINE - READ LESSON TABLE INTO MEMORY
675 QNAME$=TNAME$
          :QINST$=STRING$(20,32)+"Loading lesson table...ple
           ase wait."
          :QPAGE$=""
          :QYTPE$=""
          :GOSUB 125
680 LOCATE 24,1
685 OPEN TABLE$ FOR INPUT AS #2
690 I=1
695 IF EOF(2) THEN 715
700 INPUT#2,PAGE%(I),START%(I),MORE%(I),TYPE$(I)
705 I=I+1
710 GOTO 695
715 CLOSE 2
720 NUMPAGES%=I-1
725 '   SORT TABLE INTO PAGE # SEQUENCE
730 FOR I=1 TO NUMPAGES%-1
735 FOR J=I+1 TO NUMPAGES%
740 IF PAGE%(I)>PAGE%(J) THEN ELSE 765
745 SWAP PAGE%(I),PAGE%(J)
750 SWAP START%(I),START%(J)
755 SWAP MORE%(I),MORE%(J)
760 SWAP TYPE$(I),TYPE$(J)
765 NEXT J
770 NEXT I
775 RETURN
780 '
785 '   SUBROUTINE - FIND CORRESPONDING TABLE SUBSCRIPT FOR
        QPAGE%
790 FOR I=1 TO NUMPAGES%
795 IF PAGE%(I)=QPAGE% THEN QMAT%=I
          :RETURN
800 NEXT I
805 QMAT%=-1
          :RETURN
810 '
815 '   SUBROUTINE - READ LESSON PAGE INTO MEMORY
820 GET 1,START%(QMAT%)
825 TYPE$=BUF1$
830 AJUMP%=VAL(BUF3$)
835 BJUMP%=VAL(BUF4$)
840 CJUMP%=VAL(BUF5$)
845 DJUMP%=VAL(BUF6$)
850 EJUMP%=VAL(BUF7$)
855 ANSWER$=BUF8$
860 MORE%=VAL(BUF9$)
865 FOR I=1 TO MORE%
870 GET 1,(START%(QMAT%)+I)
```

```
875 DUMMY$=BUF10$
880 TEXT$(I)=BUF11$
885 NEXT I
890 RETURN
895 '
900 '   SUBROUTINE - INPUT LESSON SCREEN
905 '   GET PAGE NUMBER
910 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=QDATA$
          :GOSUB 125
915 LOCATE 4,16
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 4,16
920 INPUT"Enter page number: ",PAGE
          :IF PAGE>9999 THEN 915 ELSE PAGE%=PAGE
925 IF PAGE%=9999 OR PAGE%=0 THEN RETURN
930 '   IF FIRST SCREEN, ENSURE NUMBERED AS PAGE 1
935 IF REC=1 AND PAGE%=1 OR REC>1 THEN 955
940 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
945 LOCATE 4,16
          :PRINT"The first page of a lesson must be page 1!"
           ;
950 LOCATE 24,1
          :GOSUB 195
          :GOTO 905
955 '   ENSURE PAGE NUMBER IN LEGAL LIMITS
960 IF PAGE%>=1 AND PAGE%<=9999 THEN 980
965 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
970 LOCATE 4,16
          :PRINT PAGE%;" is not a valid page number!";
975 LOCATE 24,1
          :GOSUB 195
          :GOTO 905
980 '   ENSURE DUPLICATE PAGE NUMBER NOT BEING ENTERED
985 FOR I=1 TO NUMPAGES%
990 IF PAGE%=PAGE%(I) THEN 1010
995 NEXT I
1000 NUMPAGES%=NUMPAGES%+1
1005 GOTO 1030
```

```
1010 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
1015 LOCATE 4,6
          :PRINT "Page #";PAGE%;" has already been used
           ! Do not duplicate page numbers!";
1020 LOCATE 24,1
          :GOSUB 195
          :GOTO 905
1025 '
1030 '   GET PAGE TYPE TO ENTER
1035 QNAME$=TNAME$
          :QTYPE$=""
          :QPAGE$=STR$(PAGE%)
          :QINST$=PROMPT$
          :GOSUB 125
1040 LOCATE 4,16
          :PRINT"<A> Input text page";
1045 LOCATE 5,16
          :PRINT"<B> Input multiple choice question page";
1050 LOCATE 6,16
          :PRINT"<C> Input true/false question page";
1055 LOCATE 24,1
          :LEGAL$="ABCabc"
          :GOSUB 150
1060 IF A$="A" THEN TYPE$="#"
          :QTYPE$=TP$
1065 IF A$="B" THEN TYPE$="?"
          :QTYPE$=MCQP$
1070 IF A$="C" THEN TYPE$="&"
          :QTYPE$=TFQP$
1075 QNAME$=TNAME$
          :QPAGE$=STR$(PAGE%)
          :QINST$=QEDIT$
          :GOSUB 125
1080 GOSUB 245
1085 '
1090 '   GET ADDITIONAL DATA REQUIRED FOR EACH PAGE TYPE
1095 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT QDATA$;
1100 IF TYPE$="#" THEN 1115
1105 IF TYPE$="?" THEN 1150
1110 IF TYPE$="&" THEN 1255
1115 '   GET ADDITIONAL DATA FOR TEXT PAGE
1120 LOCATE 21,1
          :PRINT STRING$(81-POS(0),32);
1125 LOCATE 22,1
          :PRINT STRING$(81-POS(0),32);
```

```
1130 LOCATE 21,21
         :PRINT STRING$(81-POS(0),32);
         :LOCATE 21,21
1135 INPUT"Jump-to page: ",AJUMP
         :IF AJUMP>9999 THEN 1130 ELSE AJUMP%=AJUMP
         :PTEST%=AJUMP%
         :GOSUB 225
1145 GOTO 1315
1150 '   GET ADDITIONAL DATA FOR MULTIPLE CHOICE QUESTION
        PAGE
1155 FOR I=16 TO 22
         :LOCATE I,1
         :PRINT STRING$(81-POS(0),32);
         :NEXT I
1160 LOCATE 16,21:PRINT"Correct answer: ";
         :LEGAL$="ABCDEabcde"
         :GOSUB 150
         :PRINT A$;
         :ANSWER$=A$
1165 LOCATE 17,21
         :PRINT STRING$(81-POS(0),32);
         :LOCATE 17,21
1170 INPUT"A-jump page   : ",AJUMP
         :IF AJUMP>9999 THEN 1165 ELSE AJUMP%=AJUMP
         :PTEST%=AJUMP%
         :GOSUB 225
1175 IF FLAG%=1 THEN 1165
1180 LOCATE 18,21
         :PRINT STRING$(81-POS(0),32);
         :LOCATE 18,21
1185 INPUT"B-jump page   : ",BJUMP
         :IF BJUMP>9999 THEN 1180 ELSE BJUMP%=BJUMP
         :PTEST%=BJUMP%
         :GOSUB 225
1190 IF FLAG%=1 THEN 1180
1195 LOCATE 19,21
         :PRINT STRING$(81-POS(0),32);
         :LOCATE 19,21
1200 INPUT"C-jump page   : ",CJUMP
         :IF CJUMP>9999 THEN 1195 ELSE CJUMP%=CJUMP
         :PTEST%=CJUMP%
         :GOSUB 225
1205 IF FLAG%=1 THEN 1195
1210 LOCATE 20,21
         :PRINT STRING$(81-POS(0),32);
         :LOCATE 20,21
1215 INPUT"D-jump page   : ",DJUMP
         :IF DJUMP>9999 THEN 1210 ELSE DJUMP%=DJUMP
         :PTEST%=DJUMP%
         :GOSUB 225
1220 IF FLAG%=1 THEN 1210
```

```
1225 LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
1230 INPUT"E-jump page   : ",EJUMP
          :IF EJUMP>9999 THEN 1225 ELSE EJUMP%=EJUMP
          :PTEST%=EJUMP%
          :GOSUB 225
1235 IF FLAG%=1 THEN 1225
1240 TYPE$="?"
1245 GOTO 1315
1250 '
1255 '   GET ADDITIONAL DATA FOR TRUE/FALSE QUESTION PAGE
1260 FOR I=18 TO 22
          :LOCATE I,1
          :PRINT STRING$(81-POS(0),32);
          :NEXT I
1265 LOCATE 19,21
          :PRINT"Correct answer: ";
          :LEGAL$="TFtf"
          :GOSUB 150
          :PRINT A$;
          :ANSWER$=A$
1270 LOCATE 20,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 20,21
1275 INPUT"T-jump page   : ",AJUMP
          :IF AJUMP>9999 THEN 1270 ELSE AJUMP%=AJUMP
          :PTEST%=AJUMP%
          :GOSUB 225
1280 IF FLAG%=1 THEN 1270
1285 LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
1290 INPUT"F-jump page   : ",BJUMP
          :IF BJUMP>9999 THEN 1285 ELSE BJUMP%=BJUMP
          :PTEST%=BJUMP%
          :GOSUB 225
1295 IF FLAG%=1 THEN 1285
1300 TYPE$="?"
1305 GOTO 1315
1310 '
1315 '   GET ALL OTHER PAGE DATA AND WRITE HEADER RECORD TO
         DISK
1320 MORE%=LC%
1325 LSET BUF1$=TYPE$
1330 LSET BUF2$=STR$(PAGE%)
1335 LSET BUF3$=STR$(AJUMP%)
1340 LSET BUF4$=STR$(BJUMP%)
1345 LSET BUF5$=STR$(CJUMP%)
1350 LSET BUF6$=STR$(DJUMP%)
1355 LSET BUF7$=STR$(EJUMP%)
1360 LSET BUF8$=ANSWER$
1365 LSET BUF9$=STR$(MORE%)
```

```
1370 LSET DUMMY$=""
1375 START%=REC
1380 PUT 1,REC
1385 '
1390 '   WRITE TEXT RECORDS TO DISK
1395 FOR I=1 TO MORE%
1400 REC=REC+1
1405 LSET BUF10$=" "
1410 LSET BUF11$=TEXT$(I)
1415 PUT 1,REC
1420 NEXT I
1425 REC=REC+1
1430 '
1435 '   UPDATE LESSON TABLE
1440 PAGE%(NUMPAGES%)=PAGE%
          :START%(NUMPAGES%)=START%
          :MORE%(NUMPAGES%)=MORE%
          :TYPE$(NUMPAGES%)=TYPE$
1445 '   RETURN TO GET NEXT PAGE
1450 ERASE TEXT$
          :DIM TEXT$(20)
1455 GOTO 905
1460 '
1465 '   SUBROUTINE - GET DISK FILENAME AND PRINT ERRORS
1470 '   PRINT DISK INSTRUCTIONS
1475 QTYPE$=""
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
1480 LOCATE 4,11
          :PRINT"Insert the disk containing the lesson file
           s in drive A";
1485 LOCATE 24,1
1490 GOSUB 195
1495 '
1500 QTYPE$=""
          :QPAGE$=""
          :QINST$=QDATA$
          :GOSUB 125
1505 LOCATE 4,6
          :PRINT"Enter lesson name (maximum of 8 characters
           ; do not include";
          :LOCATE 5,6
1510 INPUT"extension; <ENTER> to abort): ",TNAME$
1515 IF TNAME$="" THEN CLOSE
          :ON ERROR GOTO 17000
          :GOTO 11000
1520 FOR I=1 TO LEN(TNAME$)
1525 TEST%=ASC(MID$(TNAME$,I,1))
1530 IF TEST%>=97 AND TEST%<=122 THEN MID$(TNAME$,I,1)=CHR$(
     TEST%-32)
1535 NEXT I
1540 '
```

```
1545 '   CHECK FOR NEEDED FILES ON DISK
1550 TEXT$="A:"+TNAME$+".TXT"
1555 TABLE$="A:"+TNAME$+".TAB"
1560 STUFILE$="A:"+TNAME$+".STU"
1565 ON ERROR GOTO 1590
1570 IF INSTR(NEED$,"A")<>0 THEN QTEST$=TEXT$
        :OPEN QTEST$ FOR INPUT AS #2
        :CLOSE #2
1575 IF INSTR(NEED$,"B")<>0 THEN QTEST$=TABLE$
        :OPEN QTEST$ FOR INPUT AS #2
        :CLOSE #2
1580 IF INSTR(NEED$,"C")<>0 THEN QTEST$=STUFILE$
        :OPEN QTEST$ FOR INPUT AS #2
        :CLOSE #2
1585 CLOSE
        :ON ERROR GOTO 17000
        :RETURN
1590 '
1595 QNAME$="ERROR"
        :QPAGE$=""
        :QINST$=ENTER$
        :GOSUB 125
1600 IF ERR=53 THEN LOCATE 4,16
        :PRINT QTEST$;" is not on this disk!";
        :GOTO 1620
1605 IF ERR=57 THEN LOCATE 4,16
        :PRINT"A device input/output error has occurred!"
        ;
        :GOTO 1620
1610 IF ERR=64 THEN PRINT@(3,15),TNAME$;" is not a valid les
     son name!";
        :GOTO 1620
1615 LOCATE 4,16
        :PRINT"An unknown disk error has occurred!";
1620 RESUME 1625
1625 LOCATE 24,1
1630 GOSUB 195
1635 GOTO 1500
1640 CLOSE
        :ON ERROR GOTO 17000
        :RETURN
1645 '
1650 '   SUBROUTINE - DISPLAY LESSON PAGE
1655 FOR I=1 TO MORE%(QMAT%)
1660 PRINT TEXT$(I);
1665 NEXT I
1670 RETURN
1675 '
10000 '********************************************
10005 '*   CONSTANT TABLE AND DEFINED FUNCTIONS     *
10010 '*         (LINES 10000-10999)                *
10015 '********************************************
10020 '
```

```
10025 OPTION BASE 1
          :KEY OFF
          :ON ERROR GOTO 17000
10030 DASH$=STRING$(80,45)
10035 PROMPT$=STRING$(22,32)+"Press <letter> of your choice.
      "
10040 QDATA$=STRING$(14,32)+"Enter requested data and press
      <ENTER> to continue."
10045 QEDIT$=STRING$(10,32)+"Enter Text"+STRING$(10,32)+"Ctr
      l<E> to End"+STRING$(10,32)+"Ctrl<Q> to Quit"
10050 MCQP$="Multiple Choice Question Page"
10055 TP$=STRING$(10,32)+"Text Page"
10060 TFQP$=STRING$(2,32)+"True/False Question Page"
10065 ENTER$=STRING$(22,32)+"Press <ENTER> to continue."
10070 YN$=STRING$(10,32)+"<Y>es     <N>o"+STRING$(25,32)
10075 CHAR8$=CHR$(29)+CHR$(32)+CHR$(29)
11000 '*********************************************
11005 '*             MAIN PROGRAM                  *
11010 '*          (LINES 11000-39999)              *
11015 '*********************************************
11020 '
11025 '   INITIALIZE COUNTERS
11030 NUMPAGES%=0
11035 '
11040 '   PRINT MAIN MENU AND ROUTE TO PROPER PORTION OF
          PROGRAM
11045 QNAME$="MAIN MENU"
          :QPAGE$=""
          :QINST$=PROMPT$
          :QTYPE$=""
          :GOSUB 125
11050 LOCATE 4,16
          :PRINT"<A>  Create a lesson file."
11055 LOCATE 5,16
          :PRINT"<B>  Edit a lesson file."
11060 LOCATE 6,16
          :PRINT"<C>  Print a lesson file."
11065 LOCATE 7,16
          :PRINT"<D>  Print a student file report."
11070 LOCATE 8,16
          :PRINT"<E>  Exit WRITE program."
11075 LOCATE 24,1
11080 LEGAL$="ABCDEabcde"
          :GOSUB 150
11085 IF A$="A" THEN 12000 ELSE IF A$="B" THEN 13000 ELSE IF
       A$="C" THEN 14000 ELSE IF A$="D" THEN 15000 ELSE IF A
      $="E" THEN 16000
      $="E" THEN 16000
11090 '
12000 '   CREATE A LESSON FILE
12005 DIM TEXT$(20),PAGE%(200),MORE%(200),TYPE$(200),START%(
      200)
12010 '   PRINT DISK INSTRUCTIONS
```

```
12015 QNAME$="CREATE A LESSON FILE"
        :QTYPE$=""
        :QPAGE$=""
        :QINST$=ENTER$
        :GOSUB 125
12020 LOCATE 4,16
        :PRINT"Insert a properly formatted disk in drive
         A."
12025 LOCATE 24,1
        :GOSUB 195
12030 '
12035 '   GET LESSON FILE NAME
12040 QNAME$="CREATE A LESSON FILE"
        :QTYPE$=""
        :QPAGE$=""
        :QINST$=QDATA$
        :GOSUB 125
12045 LOCATE 4,6
        :PRINT"Enter lesson name (maximum of 8 characters;
          do not include";
        :LOCATE 5,6
12050 INPUT"extension; <ENTER> to abort): ",TNAME$
12055 IF TNAME$="" THEN GOTO 11000
12060 FOR I=1 TO LEN(TNAME$)
12065 TEST%=ASC(MID$(TNAME$,I,1))
12070 IF TEST%>=97 AND TEST%<=122 THEN MID$(TNAME$,I,1)=CHR$
      (TEST%-32)
12075 NEXT I
12080 TEXT$="A:"+TNAME$+".TXT"
12085 TABLE$="A:"+TNAME$+".TAB"
12090 '
12095 '   OPEN LESSON FILE BUFFER, INITIALIZE RECORD
          COUNTER, AND START
12100 '   PAGE INPUT
12105 ON ERROR GOTO 12180
12110 OPEN TEXT$ AS #1 LEN=81
12115 FIELD #1,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5
       AS BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,
      46 AS DUMMY$
12120 FIELD #1,1 AS BUF10$,80 AS BUF11$
12125 ON ERROR GOTO 17000
12130 REC=1
12135 GOSUB 900
12140 CLOSE
12145 '
12150 '   CREATE LESSON TABLE FILE
12155 QNAME$=TNAME$
        :QYPTE$=""
        :QPAGE$=""
        :QINST$=STRING$(14,32)+"Writing lesson files to d
         isk...please wait."
        :GOSUB 125
12160 GOSUB 560
```

```
12165 CLOSE
        :ERASE TEXT$,PAGE%,MORE%,TYPE$,START%
12170 CLS
        :GOTO 11025   '   RETURN TO MAIN MENU
12175 '
12180 '   ERROR HANDLING ROUTINE FOR FILE NAME TO CREATE
12185 QNAME$="ERROR"
        :QTYPE$=""
        :QPAGE$=""
        :QINST$=ENTER$
        :GOSUB 125
12190 IF ERR=57 THEN LOCATE 4,16
        :PRINT"A device input/output error has occurred!"
        :GOTO 12205
12195 IF ERR=64 THEN LOCATE 4,16
        :PRINT TNAME$;" is not a valid lesson name!";
        :GOTO 12205
12200 LOCATE 4,16
        :PRINT"An unknown disk error has occurred!";
12205 LOCATE 24,1
        :GOSUB 195
12210 RESUME 12215
12215 CLOSE
        :ON ERROR GOTO 17000
        :GOTO 12015
13000 '   EDIT A LESSON FILE
13005 QNAME$="EDIT A LESSON FILE"
13010 '   GET LESSON FILE NAME
13015 NEED$="AB"
        :GOSUB 1465
13020 DIM TEXT$(20),PAGE%(200),START%(200),MORE%(200),TYPE$(
      200)
13025 '
13030 '   READ TABLE FILE INTO MEMORY
13035 GOSUB 670
13040 CLOSE
        :OPEN TEXT$ AS #1 LEN=81
13045 FIELD #1,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5
       AS BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,
      46 AS DUMMY$
13050 FIELD #1,1 AS BUF10$,80 AS BUF11$
13055 '
13060 '   DISPLAY EDIT MAIN MENU
13065 QNAME$=TNAME$
        :QTYPE$=""
        :QPAGE$=""
        :QINST$=PROMPT$
        :GOSUB 125
13070 LOCATE 4,16
        :PRINT"<A>  Add screen to lesson."
13075 LOCATE 5,16
        :PRINT"<B>  Delete screen from lesson."
```

```
13080 LOCATE 6,16
        :PRINT"<C>  Modify existing screen."
13085 LOCATE 7,16
        :PRINT"<D>  Return to WRITE main menu."
13090 LOCATE 24,1
13095 LEGAL$="ABCDabcd"
        :GOSUB 150
13100 IF A$="A" THEN 13105 ELSE IF A$="B" THEN 13130 ELSE IF
       A$="C" THEN 13270 ELSE IF A$="D" THEN 13700
13105 '   ADD SCREEN TO LESSON
13110 '   GET CURRENT LAST RECORD
13115 REC=(LOF(1)/81)+1
13120 GOSUB 900
13125 GOTO 13060
13130 '  DELETE PAGE FROM EXISTING LESSON FILE
13135 '   GET PAGE NUMBER
13140 QNAME$=TNAME$
        :QTYPE$="DELETE A PAGE"
        :QPAGE$=""
        :QINST$=QDATA$
        :GOSUB 125
13145 LOCATE 4,16
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 4,16
13150 INPUT"Enter page number to delete: ",PAGE
        :IF PAGE>9999 THEN 13145 ELSE PAGE%=PAGE
13155 '   FIND TABLE SUBSCRIPT/READ LESSON PAGE INTO MEMORY
13160 QPAGE%=PAGE%
        :GOSUB 785
13165 IF QMAT%=-1 THEN LOCATE 6,16
        :PRINT"Page not found in lesson table!";
        :LOCATE 24,1
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 24,1
        :PRINT ENTER$;
        :LOCATE 24,1
        :GOSUB 195
        :GOTO 13060
13170 GOSUB 815
13175 '   DISPLAY PAGE
13180 QNAME$=TNAME$
        :QTYPE$="DELETE A PAGE"
        :QPAGE$=STR$(PAGE%)
        :QINST$=STRING$(15,32)+"<D> to Delete Page"+STRING
         $(15,32)+"<Q> to Abort"
        :GOSUB 125
13185 GOSUB 1650
13190 '   GET DELETE DECISION
13195 LOCATE 24,1
13200 LEGAL$="QqDd"
        :GOSUB 150
```

```
13205 IF A$="Q" THEN ERASE TEXT$
          :DIM TEXT$(20)
          :GOTO 13060
13210 FOR REC=START%(QMAT%) TO START%(QMAT%)+MORE%(QMAT%)
13215 LSET BUF10$="*"
13220 LSET BUF11$=STRING$(80,42)
13225 PUT 1,REC
13230 NEXT REC
13235 '   UPDATE LESSON TABLE
13240 FOR I=QMAT% TO NUMPAGES%
13245 PAGE%(I)=PAGE%(I+1)
          :START%(I)=START%(I+1)
          :MORE%(I)=MORE%(I+1 )
          :TYPE$(I)=TYPE$(I+1 )
          :NEXT I
13250 NUMPAGES%=NUMPAGES%-1
13255 ERASE TEXT$
          :DIM TEXT$(20)
13260 GOTO 13060
13265 '
13270 '  MODIFY EXISTING PAGE
13275 '   GET PAGE NUMBER
13280 QNAME$=TNAME$
          :QTYPE$="MODIFY A PAGE"
          :QPAGE$=""
          :QINST$=QDATA$
          :GOSUB 125
13285 LOCATE 4,16
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 4,16
13290 INPUT"Enter page number to modify: ",PAGE
          :IF PAGE>9999 THEN 13285 ELSE PAGE%=PAGE
13295 '   FIND TABLE SUBSCRIPT/READ LESSON PAGE INTO MEMORY
13300 QPAGE%=PAGE%
          :GOSUB 785
13305 IF QMAT%=-1 THEN LOCATE 6,16
          :PRINT"Page not found in lesson table!";
          :LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);:LOCATE 24,1
          :PRINT ENTER$;
          :LOCATE 24,1
          :GOSUB 195
          :GOTO 13060
13310 GOSUB 815
13315 '   DISPLAY PAGE
13320 QNAME$=TNAME$
          :QTYPE$="MODIFY A PAGE"
          :QPAGE$=STR$(PAGE%)
          :QINST$="   Ctrl <E>nd text edit   Ctrl <Q>uit and
            cancel   Ctrl <D>elete character"
          :GOSUB 125
13325 GOSUB 1650
13330 LOCATE 3,1
```

```
13335 A$=INKEY$
13340 IF A$="" THEN 13335
13345 IF ASC(A$)=17 THEN ERASE TEXT$
        :DIM TEXT$(20)
        :GOTO 13060   '   QUIT EDITOR
13350 IF ASC(A$)=5 THEN GOTO 13390   '   END TEXT EDITING
13355 IF A$=CHR$(0)+CHR$(72) THEN IF CSRLIN=3 THEN 13335 ELS
      E LOCATE CSRLIN-1,POS(0)
        :GOTO 13335   '   UP ARROW
13360 IF A$=CHR$(0)+CHR$(80) THEN IF CSRLIN=MORE%+2 THEN 133
      35 ELSE LOCATE CSRLIN+1,POS(0)
        :GOTO 13335   '   DOWN ARROW
13365 IF A$=CHR$(8) OR A$=CHR$(0)+CHR$(75) THEN IF POS(0)=1
      THEN 13335 ELSE LOCATE CSRLIN,POS(0)-1
        :GOTO 13335   '   LEFT ARROW
13370 IF A$=CHR$(0)+CHR$(77) THEN IF POS(0)=80 THEN 13335 EL
      SE LOCATE CSRLIN,POS(0)+1
        :GOTO 13335   '   RIGHT ARROW
13375 IF A$=CHR$(4) OR A$=CHR$(0)+CHR$(83) THEN TPOS=POS(0)
        :TROW=CSRLIN
        :TEXT$(TROW-2)=LEFT$(TEXT$(TROW-2),TP OS-1)+RIGHT$
         (TEXT$(TROW-2),80-TPOS)+CHR$(32)
        :LOCATE TROW,1
        :PRINT TEXT$(TROW-2);
        :LOCATE TROW,TPOS
        :GOTO 13335   '   DELETE CHARACTER
13380 IF ASC(A$)>=32 AND ASC(A$)<=126 THEN MID$(TEXT$(CSRLIN
      -2),POS(0),1)=A$
        :PRINT A$;
        :IF CSRLIN=MORE%+3 AND POS(0)=1 THEN PRINT CHAR8$;
        :GOTO 13335   '   OVERSTRIKE WITH VALID ASCII
         CHARACTER
13385 GOTO 13335   '   INVALID KEY ENTRY
13390 '   END TEXT EDITING - CHANGE ADDITIONAL DATA?
13395 '   CHANGE PAGE NUMBER?
13400 FOR I=20 TO 22
        :LOCATE I,1
        :PRINT STRING$(81-POS(0),32);
        :NEXT I
13405 LOCATE 21,21
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,21
        :PRINT"Page number: ";PAGE%;
        :LOCATE 24,1
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 24,1
        :PRINT" Change page number?";YN$;
        :LOCATE 24,1
13410 LEGAL$="YyNn"
        :GOSUB 150
13415 IF A$<>"Y" THEN 13460
```

```
13420 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT QDATA$;
13425 LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
          :PRINT"New page number: ";
13430 INPUT"",PAGE
          :IF PAGE>9999 THEN 13425
13435 FOR I=1 TO NUMPAGES%
          :IF PAGE=PAGE%(I) THEN 13445 ELSE NEXT I
13440 PAGE%=PAGE
          :PAGE%(QMAT%)=PAGE%
          :GOTO 13405
13445 LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
          :PRINT PAGE;" already used!";
13450 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT ENTER$;
          :LOCATE 24,1
          :GOSUB 195
13455 GOTO 13405
13460 '   ASSIGN TEMPORARY TYPE$ TO T/F QUESTION
13465 IF TYPE$="?" AND (ANSWER$="T" OR ANSWER$="F")  THEN TY
      PE$="&"
13470 IF TYPE$="#" THEN 13485
13475 IF TYPE$="?" THEN 13510
13480 IF TYPE$="&" THEN 13595
13485 '   TEXT PAGE - CHANGE JUMP PAGE?
13490 LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
          :PRINT "Jump page: ";AJUMP%;
13495 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT" Change jump page?";YN$;
          :LOCATE 24,1
          :LEGAL$="YyNn"
          :GOSUB 150
```

```
13500 IF A$<>"Y" THEN 13670 ELSE LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT QDATA$;
          :LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
          :PRINT"Jump page: ";
          :INPUT"",AJUMP
          :IF AJUMP<10000 AND AJUMP>0 THEN AJUMP%=AJUMP ELSE
           GOTO 13500
13505 GOTO 13485
13510 '   MULTIPLE CHOICE QUESTION
13515 '   CHANGE CORRECT ANSWER?
13520 LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
          :PRINT"Correct answer: ";ANSWER$;
13525 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT" Change correct answer?";YN$;
          :LOCATE 24,1
          :LEGAL$="YyNn"
          :GOSUB 150
13530 IF A$<>"Y" THEN 13540
13535 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT PROMPT$;
          :LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
          :PRINT"Correct answer: ";
          :LEGAL$="ABCDEabcde"
          :GOSUB 150
          :PRINT A$;
          :ANSWER$=A$
          :GOTO 13520
13540 '   CHANGE ANSWER JUMP PAGES?
13545 LOCATE 21,1
          :PRINT"    AJUMP: ";AJUMP%;"    BJUMP: ";BJUMP%;"
              CJUMP: ";CJUMP%;"    DJUMP: ";DJUMP%;"    EJUM
           P: ";EJUMP%;
13550 LOCATE 24,1
          :PRINT" Change answer jump pages?";YN$;
          :LOCATE 24,1
          :LEGAL$="YyNn"
          :GOSUB 150
13555 IF A$<>"Y" THEN 13670
```

```
13560 LOCATE 24,1
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 24,1
        :PRINT PROMPT$;
        :LOCATE 24,1
        :PRINT" Which one?";
        :LOCATE 24,1
        :LEGAL$="ABCDEabcde"
        :GOSUB 150
        :LOCATE 24,1
        :PRINT QDATA$;
        :LOCATE 21,1
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,1
13565 IF A$="A" THEN LOCATE 21,21
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,21
        :PRINT"AJUMP: ";
        :INPUT"",AJUMP
        :IF AJUMP>0 AND AJUMP<10000 THEN AJUMP%=AJUMP ELSE
          GOTO 13565
13570 IF A$="B" THEN LOCATE 21,21
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,21
        :PRINT"BJUMP: ";
        :INPUT"",BJUMP
        :IF BJUMP>0 AND BJUMP<10000 THEN BJUMP%=BJUMP ELSE
          GOTO 13570
13575 IF A$="C" THEN LOCATE 21,21
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,21
        :PRINT"CJUMP: ";
        :INPUT"",CJUMP
        :IF CJUMP>0 AND CJUMP<10000 THEN CJUMP%=CJUMP ELSE
          GOTO 13575
13580 IF A$="D" THEN LOCATE 21,21
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,21
        :PRINT"DJUMP: ";
        :INPUT"",DJUMP
        :IF DJUMP>0 AND DJUMP<10000 THEN DJUMP%=DJUMP ELSE
          GOTO 13580
13585 IF A$="E" THEN LOCATE 21,21
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,21
        :PRINT"EJUMP: ";
        :INPUT"",EJUMP
        :IF EJUMP>0 AND EJUMP<10000 THEN EJUMP%=EJUMP ELSE
          GOTO 13585
13590 GOTO 13540
13595 '   TRUE/FALSE QUESTION
13600 '   CHANGE CORRECT ANSWER?
```

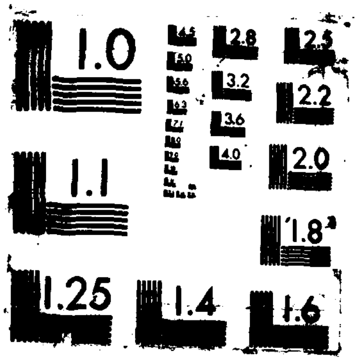
```
13605 LOCATE 21,21
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
          :PRINT"Correct answer: ";ANSWER$;
13610 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT" Change correct answer?";YN$;
          :LOCATE 24,1
          :LEGAL$="YyNn"
          :GOSUB 150
13615 IF A$<>"Y" THEN 13625
13620 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT PROMPT$;
          :LOCATE 21,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,21
          :PRINT"Correct answer: ";
          :LEGAL$="TtFf"
          :GOSUB 150
          :PRINT A$;
          :ANSWER$=A$
          :GOTO 13600
13625 '   CHANGE ANSWER JUMP PAGES?
13630 LOCATE 21,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,1
          :PRINT STRING$(24,32);"TJUMP: ";AJUMP%;"
           FJUMP: ";BJUMP%;
13635 LOCATE 24,1
          :PRINT" Change answer jump pages?";YN$;
          :LOCATE 24,1
          :LEGAL$="YyNn"
          :GOSUB 150
13640 IF A$<>"Y" THEN TYPE$="?"
          :GOTO 13670
13645 LOCATE 24,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 24,1
          :PRINT PROMPT$;
          :LOCATE 24,1
          :PRINT" Which one?";
          :LOCATE 24,1
          :LEGAL$="TtFf"
          :GOSUB 150
13650 LOCATE 21,1
          :PRINT STRING$(81-POS(0),32);
          :LOCATE 21,1
```

```
13655 IF A$="T" THEN LOCATE 21,21
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,21
        :PRINT"TJUMP: ";
        :INPUT"",AJUMP
        :IF AJUMP>0 AND AJUMP<10000 THEN AJUMP%=AJUMP ELSE
         GOTO 13655
13660 IF A$="F" THEN LOCATE 21,21
        :PRINT STRING$(81-POS(0),32);
        :LOCATE 21,21
        :PRINT"FJUMP: ";
        :INPUT"",BJUMP
        :IF BJUMP>0 AND BJUMP<10000 THEN BJUMP%=BJUMP ELSE
         GOTO 13660
13665 TYPE$="?"
        :GOTO 13630
13670 '   END EDIT - WRITE PAGE TO DISK
13675 LSET BUF1$=TYPE$
        :LSET BUF2$=STR$(PAGE%)
        :LSET BUF3$=STR$(AJUMP%)
        :LSET BUF4$=STR$(BJUMP%)
        :LSET BUF5$=STR$(CJUMP%)
        :LSET BUF6$=STR$(DJUMP%)
        :LSET BUF7$=STR$(EJUMP%)
        :LSET BUF8$=ANSWER$
        :LSET BUF9$=STR$(MORE%)
        :LSET DUMMY$=""
13680 REC=START%(QMAT%)
13685 PUT 1,REC
13690 FOR I=1 TO MORE%
        :REC=REC+1
        :LSET BUF10$=" "
        :LSET BUF11$=TEXT$(I)
        :PUT 1,REC
        :NEXT I
13695 ERASE TEXT$
        :DIM TEXT$(20)
        :GOTO 13060
13700 '  EXIT EDITOR - RETURN TO MAIN MENU
13705 QNAME$="EXIT EDITOR"
        :QPAGE$=""
        :QTYPE$=""
        :QINST$=STRING$(14,32)+"Writing lesson files to di
         sk...please wait."
        :GOSUB 125
13710 CLOSE
13715 GOSUB 560
13720 CLOSE
13725 ERASE PAGE%,START%,MORE%,TYPE$,TEXT$
13730 GOTO 11025
13735 '
14000 '   PRINT A LESSON FILE
```

```
14005 DIM TEXT$(20),PAGE%(200),MORE%(200),TYPE$(200),START%(
      200)
14010 '   GET LESSON FILE NAME
14015 QNAME$="PRINT A LESSON FILE"
14020 NEED$="AB"
14025 GOSUB 1465
14030 GOSUB 670
14035 '   ASSIGN QUESTION NUMBERS
14040 QNUM%=1
14045 FOR I=1 TO NUMPAGES%
14050 IF TYPE$(I)<>"?" THEN 14065
14055 QNUM%(I)=QNUM%
14060 QNUM%=QNUM%+1
14065 NEXT I
14070 '
14075 QNAME$=TNAME$
          :QTYPE$="PRINT A LESSON FILE"
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
14080 LOCATE 4,16
          :PRINT"Ensure printer is ready for printing.";
14085 LOCATE 24,1
14090 GOSUB 195
14095 '
14100 '   OPEN TEXT FILE
14105 OPEN TEXT$ AS #1 LEN=81
14110 FIELD #1,1 AS BUF1$,5 AS BUF2$,5 AS BUF3$,5 AS BUF4$,5
       AS BUF5$,5 AS BUF6$,5 AS BUF7$,1 AS BUF8$,3 AS BUF9$,
      46 AS DUMMY$
14115 FIELD #1,1 AS BUF10$,80 AS BUF11$
14120 '
14125 '   READ PAGES OF TEXT AND OUTPUT TO PRINTER
14130 FOR QMAT%=1 TO NUMPAGES%
14135 GOSUB 815
14140 LPRINT DASH$
14145 LPRINT
14150 LPRINT "Page # ";PAGE%(QMAT%);
14155 IF TYPE$(QMAT%)="?" THEN LPRINT "/ Question #";QNUM%(Q
      MAT%) ELSE LPRINT
14160 LPRINT
14165 FOR J=1 TO MORE%
14170 LPRINT TEXT$(J)
14175 NEXT J
14180 LPRINT
14185 IF TYPE$(QMAT%)="#" THEN LPRINT "Jump:";AJUMP%
          :GOTO 14200
14190 IF TYPE$(QMAT%)="?" AND INSTR("TF",ANSWER$)<>0 THEN LP
      RINT"T-jump:";AJUMP%;"  F-jump:";BJUMP%; TAB(69);"Corr
      ect: ";ANSWER$
          :GOTO 14200
```

```
14195 IF TYPE$(QMAT%)="?" AND INSTR("ABCDE",ANSWER$)<>0 THEN
       LPRINT"A-jump:";AJUMP%;"  B-jump:";BJUMP%;"  C-jump:"
      ;CJUMP%;"  D-jump:";DJUMP%;"  E-jump:";EJUMP%;TAB(69);
      "Correct: ";ANSWER$
14200 LPRINT
14205 ERASE TEXT$
          :DIM TEXT$(20)
14210 NEXT QMAT%
14215 LPRINT DASH$
14220 LPRINT CHR$(12);
14225 CLOSE
14230 ERASE PAGE%,START%,MORE%,TYPE$,TEXT$
14235 CLS
          :GOTO 11025    '   RETURN TO MAIN MENU
14240 '
15000 '   PRINT A STUDENT FILE REPORT
15005 QNAME$="PRINT A STUDENT FILE REPORT"
15010 NEED$="C"
          :GOSUB 1465
15015 '
15020 QNAME$=TNAME$
          :QTYPE$="PRINT A STUDENT FILE"
          :QPAGE$=""
          :QINST$=ENTER$
          :GOSUB 125
15025 LOCATE 4,16
          :PRINT"Ensure printer is ready for printing.";
15030 LOCATE 24,1
15035 GOSUB 195
15040 '
15045 '   PRINT HEADER INFORMATION
15050 LPRINT STRING$(26,32);TNAME$;" STUDENT FILE REPORT";TA
      B(65);DATE$
15055 LPRINT
15060 LPRINT DASH$
15065 LPRINT"STUDENT NAME/DATE   ASKED  RIGHT  WRONG   PCT
            WRONG QUESTIONS-RESPONSES"
15070 LPRINT DASH$
15075 LPRINT
15080 '
15085 '   PRINT STUDENT INFORMATION
15090 OPEN STUFILE$ FOR INPUT AS #1
15095 DIM WQUEST%(100),WQUEST$(100)
15100 IF EOF(1) THEN 15185
15105 INPUT#1,STUDENT$,QDATE$,TCOUNT%,RCOUNT%,WCOUNT%
15110 PCT=(RCOUNT%/TCOUNT%)*100
15115 LPRINT USING "\      \ \       \    ###    ###    ###
       ###.#";STUDENT$,QDATE$,TCOUNT%,RCOUNT%,WCOUNT%,PCT;
15120 IF WCOUNT%=0 GOTO 15170
15125 FOR I=1 TO WCOUNT%
15130 INPUT#1,WQUEST%(I),WQUEST$(I)
15135 NEXT I
15140 FOR I=1 TO WCOUNT% STEP 5
```

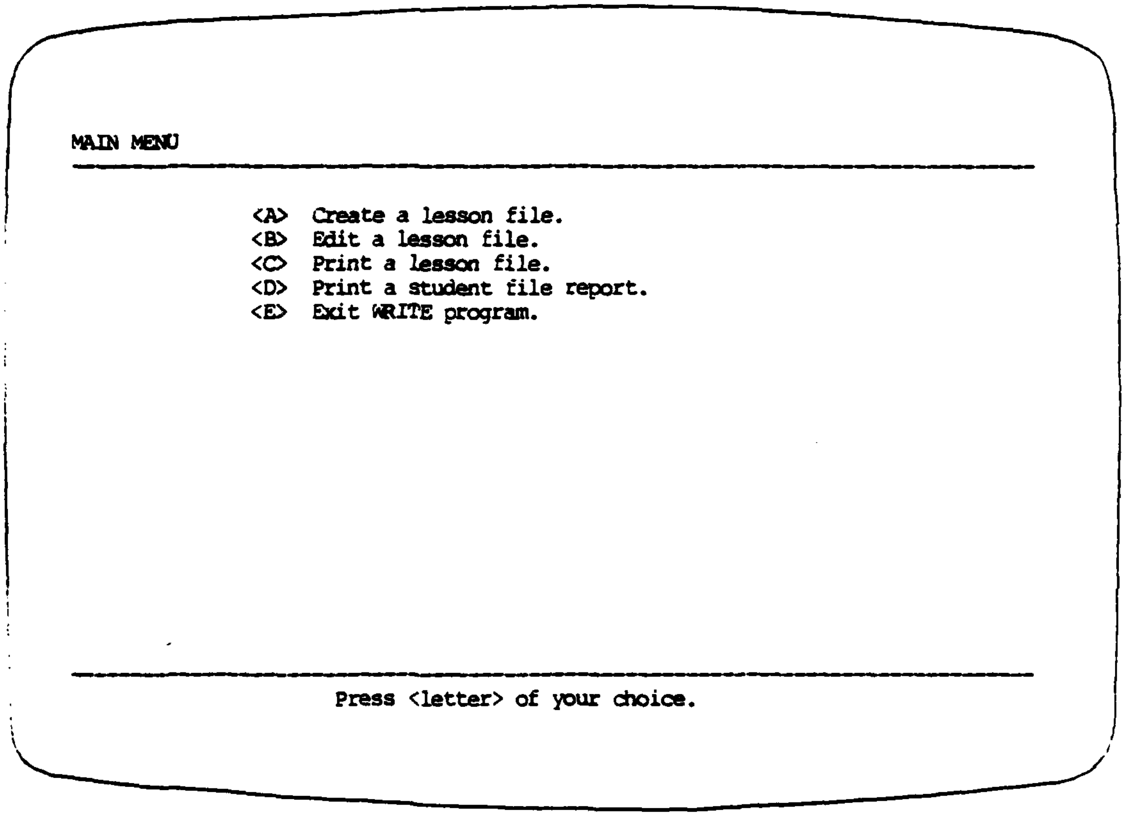
AD-A187 268   A GENERAL APPLICATION COMPUTER-ASSISTED INSTRUCTION   3/3
SYSTEM FOR MICROCOMPUTERS(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF SYST.. R MASON
UNCLASSIFIED   SEP 87 AFIT/GLM/LSR/87S-45   F/G 12/5   NL


1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

```
15145 LPRINT TAB(51);
15150 FOR J=0 TO 4
15155 IF WQUEST%(I+J)<>0 THEN LPRINT USING "###_-! ";WQUEST%
      (I+J);WQUEST$(I+J);
15160 NEXT J
15165 NEXT I
15170 LPRINT
          :LPRINT
15175 ERASE WQUEST%,WQUEST$
15180 GOTO 15095
15185 '   END OF REPORT ROUTINE
15190 LPRINT DASH$
15195 LPRINT CHR$(12);
15200 ERASE WQUEST%,WQUEST$
15205 CLOSE
          :CLS
          :GOTO 11025    '    RETURN TO MAIN MENU
15210 '
16000 '   EXIT PROGRAM ROUTINE
16005 '
16010 CLS
          :END
17000 '   PROGRAM FATAL ERROR ROUTINE
17005 STAR$=STRING$(10,32)+STRING$(60,42)
17010 CLS
17015 LOCATE 4,1
          :PRINT STAR$;
          :FOR I=5 TO 16
          :LOCATE I,11
          :PRINT"**";
          :LOCATE I,69
          :PRINT"**";
          :NEXT I
          :LOCATE 16,1
          :PRINT STAR$;
17020 LOCATE 6,22
          :PRINT"FATAL PROGRAM ERROR DURING EXECUTION";
17025 LOCATE 8,26
          :PRINT"Error code: ";ERR;" in line ";ERL;
17030 LOCATE 11,14
          :PRINT"Retain above data and refer to WRITE/BAS Us
           er's Guide";
17035 LOCATE 14,24
          :PRINT"Press <ENTER> to restart program.";
17040 LOCATE 14,23
17045 A$=INKEY$
          :IF A$<>CHR$(13) THEN 17045
17050 ' ERASE TEXT$,PAGE%,MORE%,TYPE$,START%
17055 CLOSE
          :RESUME 11000
17060 END
```

Appendix E: WRITE/BAS Courseware Example

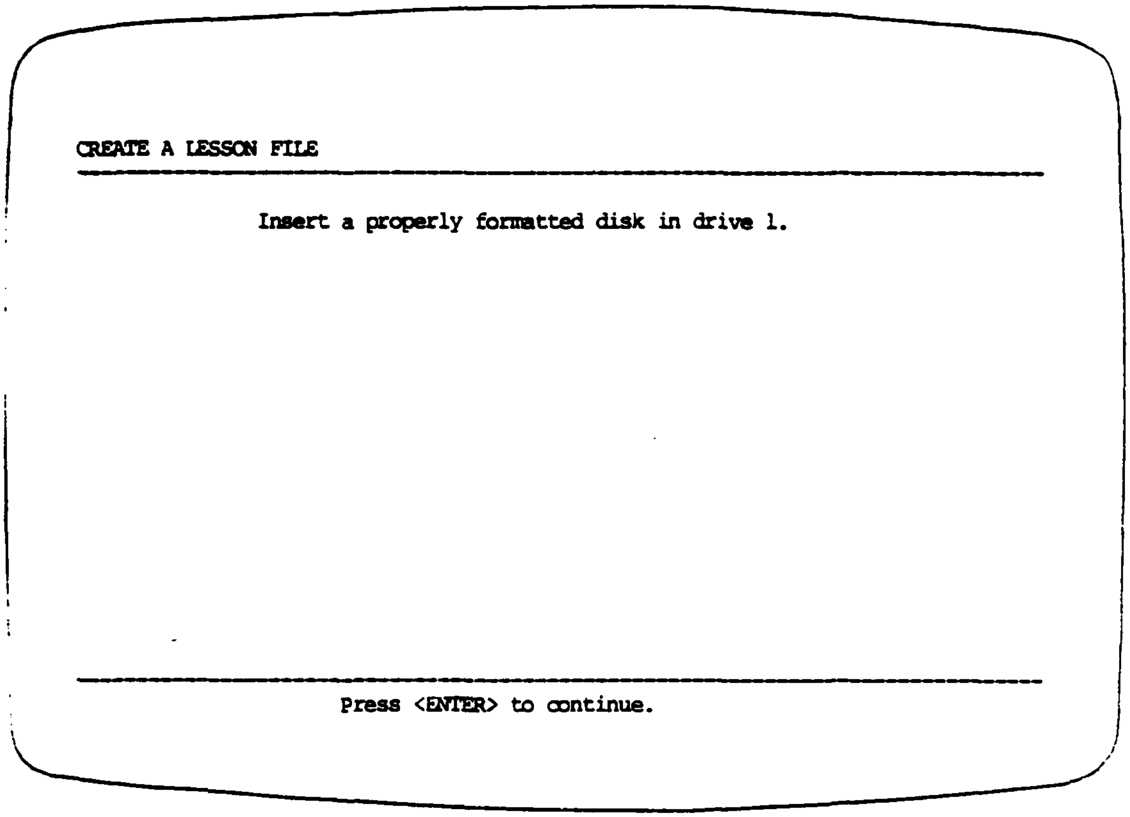
This appendix contains example courseware generated using the WRITE/BAS program. Figures E.1, E.2, E.3, and E.4 contain hardcopy prints of screens as they appear during lesson creating, editing, and printing. Although all possible screens have not been shown, a representative sample has been illustrated. Figures E.5 and E.6 illustrate the lesson and student file reports generated by the program.



```
MAIN MENU
______________________________________________________________

              <A>  Create a lesson file.
              <B>  Edit a lesson file.
              <C>  Print a lesson file.
              <D>  Print a student file report.
              <E>  Exit WRITE program.

______________________________________________________________
                    Press <letter> of your choice.
```

Initial screen of WRITE/BAS program.

Figure E.1: WRITE/BAS Main Menu



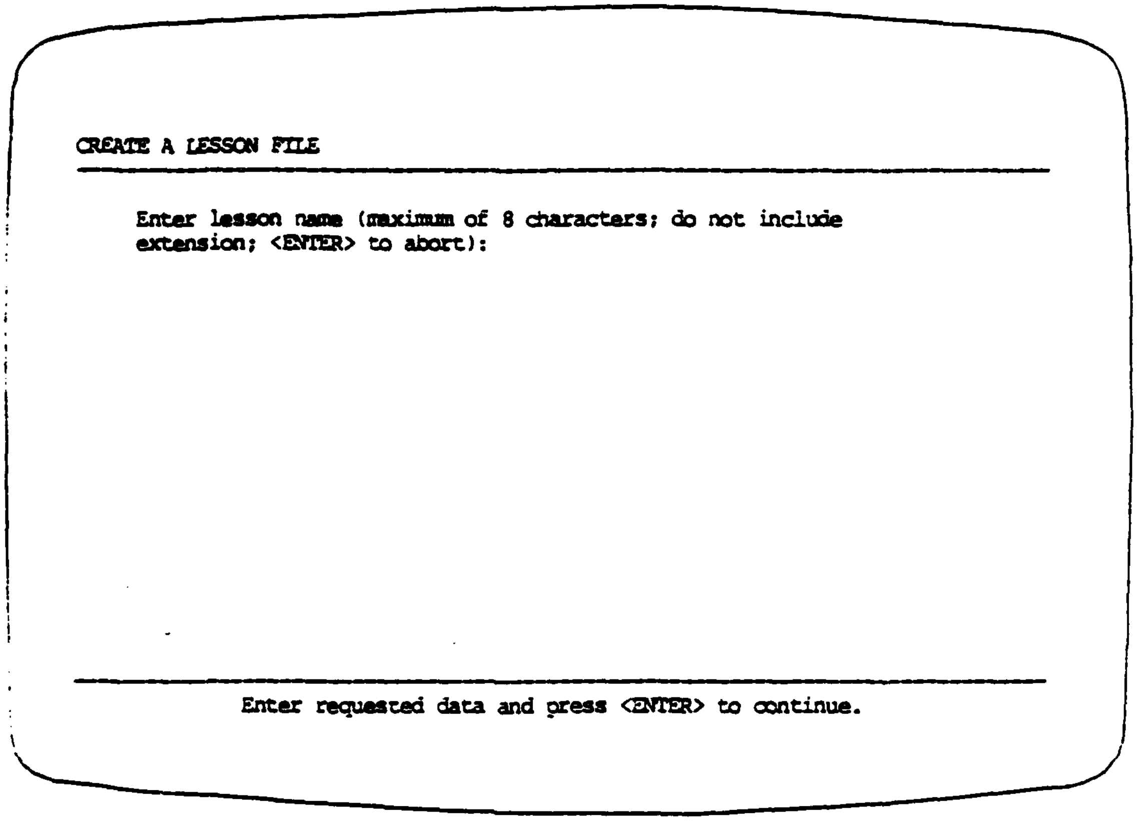
CREATE A LESSON FILE

Insert a properly formatted disk in drive 1.

Press <ENTER> to continue.

Disk instructions for creating a lesson file.

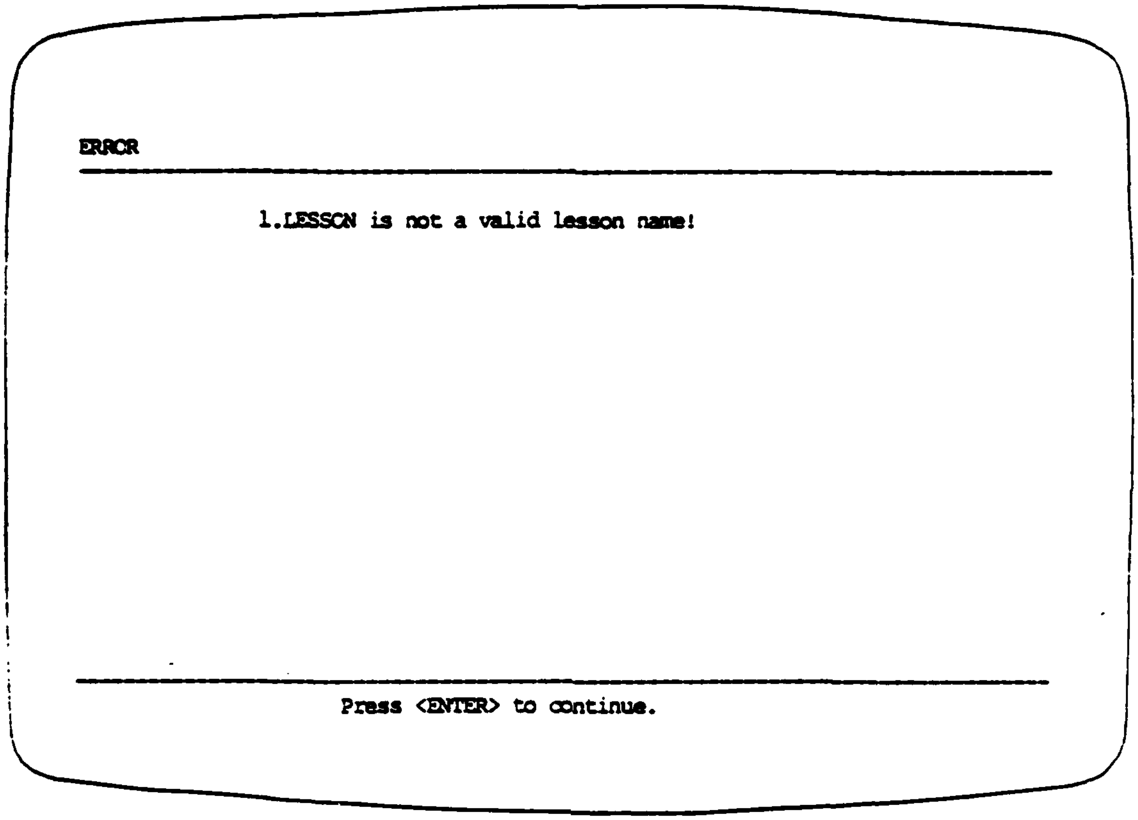
Figure E.2: Creating a Lesson File with WRITE/BAS



```
CREATE A LESSON FILE
________________________________________________________________

     Enter lesson name (maximum of 8 characters; do not include
     extension; <ENTER> to abort):









________________________________________________________________
           Enter requested data and press <ENTER> to continue.
```

Input of lesson file name for creating a lesson file.

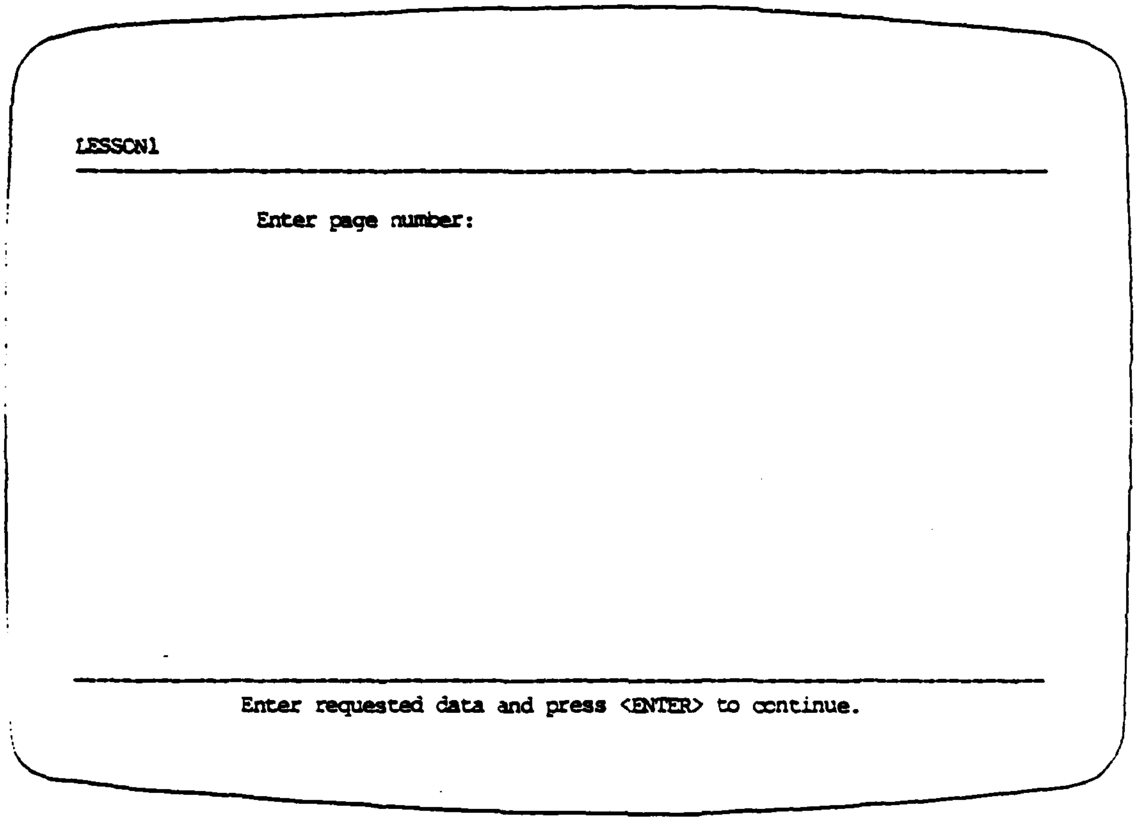
Figure E.2: Creating a Lesson File with WRITE/BAS (continued)



```
ERROR
________________________________________________________________

                 1.LESSON is not a valid lesson name!

________________________________________________________________
                    Press <ENTER> to continue.
```

Typical error message following erroneous input of lesson file name. Pressing <ENTER> returns program to enter lesson file name prompt.

Figure E.2: Creating a Lesson File with WRITE/BAS (continued)



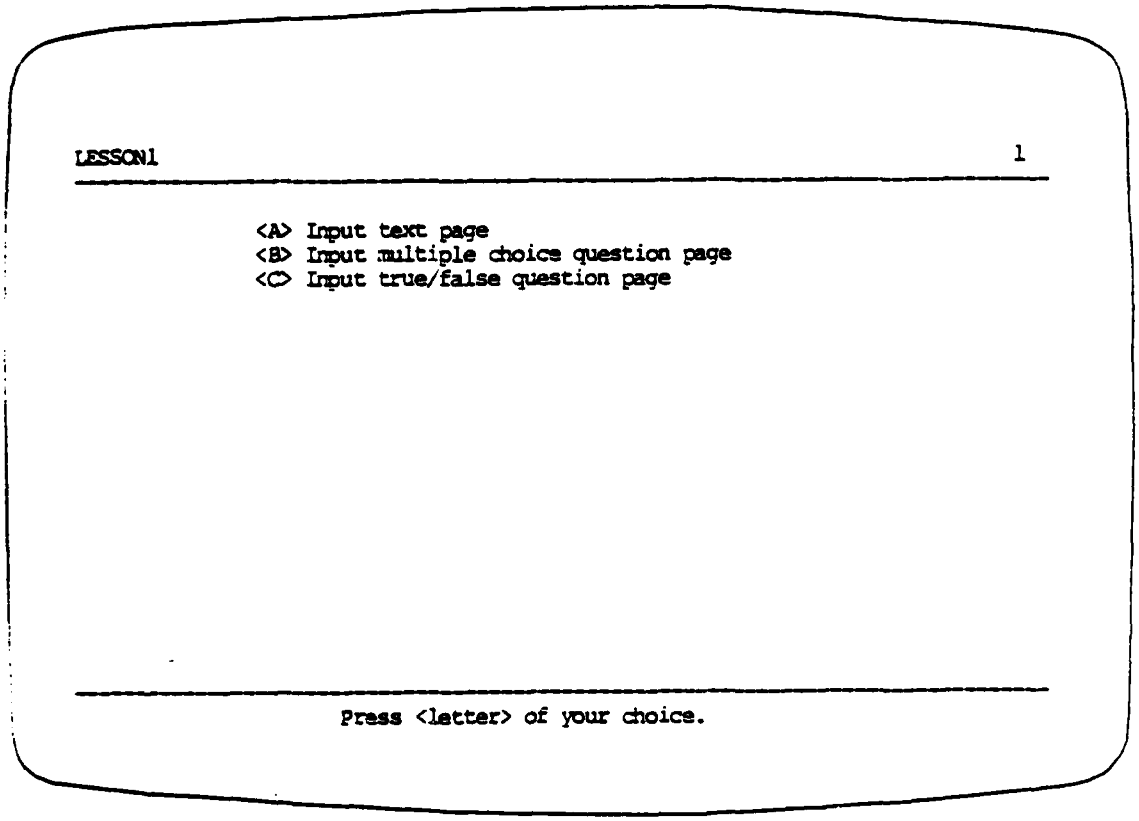
LESSON1

Enter page number:

Enter requested data and press <ENTER> to continue.

Input of lesson screen page number. First page input is validated to ensure it is numbered as page 1. Additional page numbers are validated to ensure they are valid and that page numbers are not duplicated in the lesson.

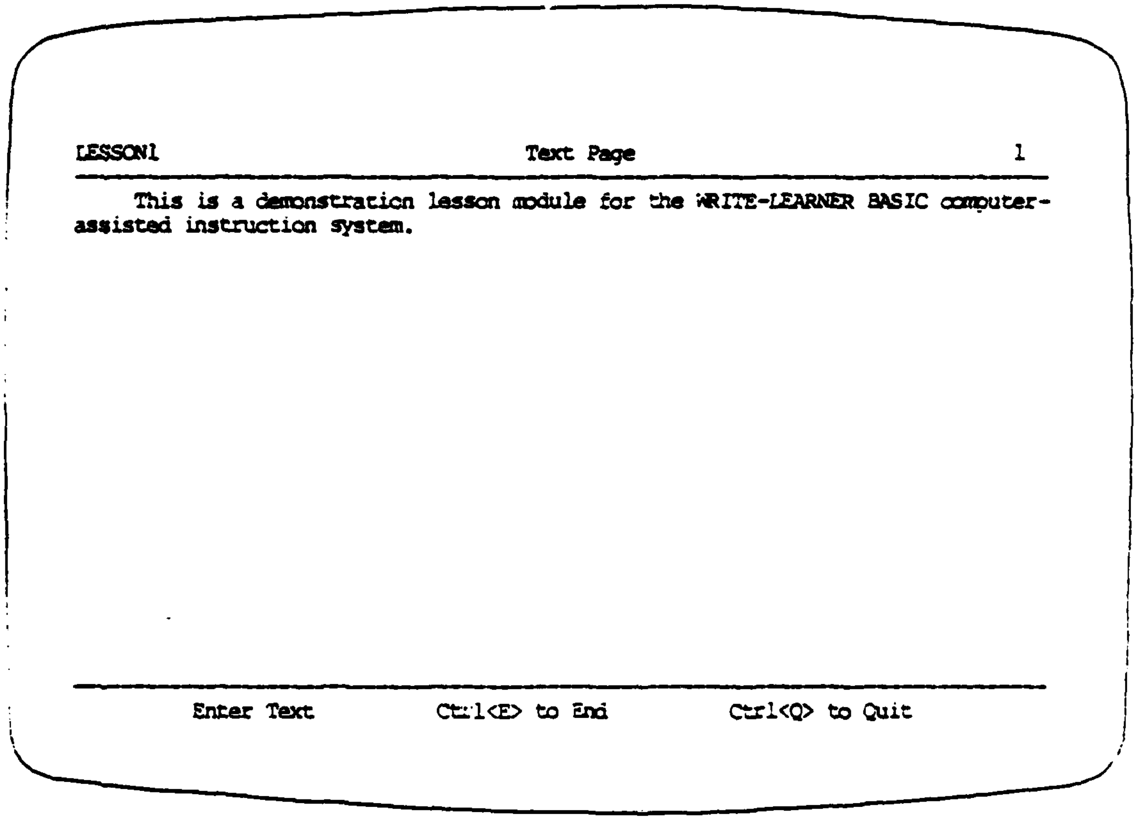
Figure E.2: Creating a Lesson File with WRITE/BAS (continued)



```
LESSON1                                                              1
______________________________________________________________________

             <A> Input text page
             <B> Input multiple choice question page
             <C> Input true/false question page

______________________________________________________________________
                    Press <letter> of your choice.
```

Input of type of page to generate.

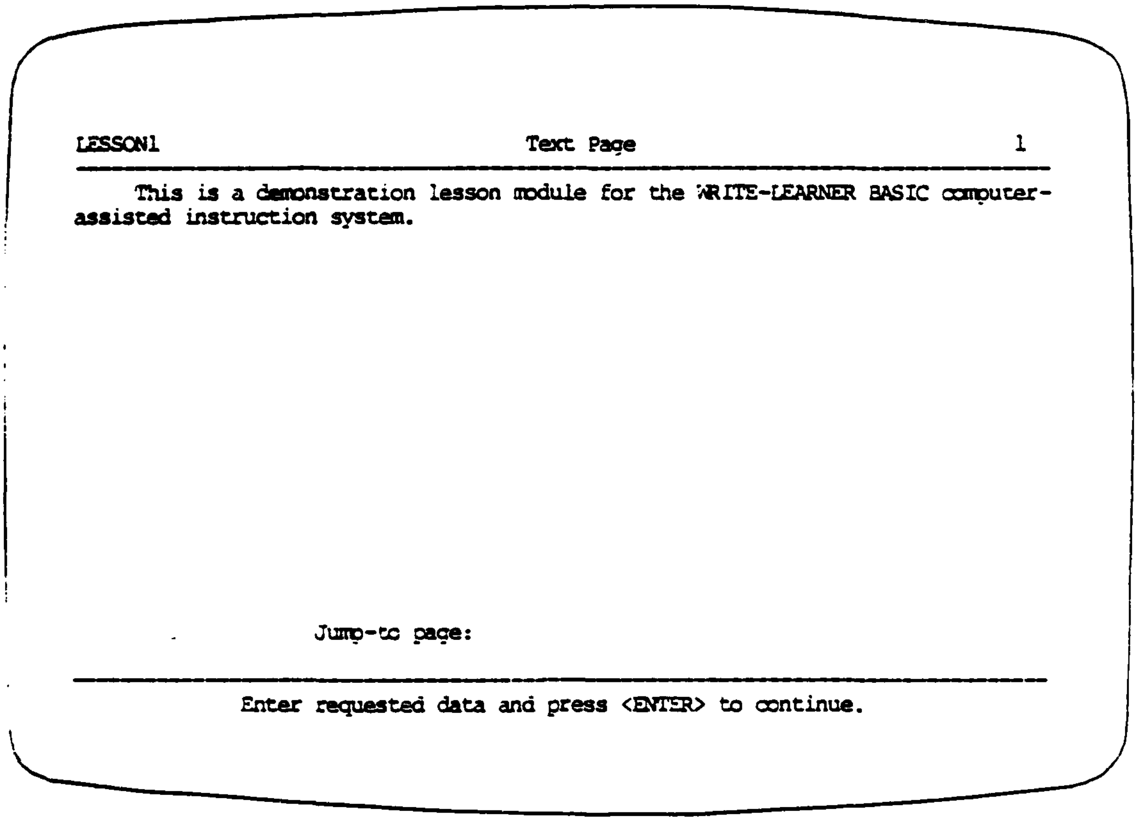
Figure E.2: Creating a Lesson File with WRITE/BAS (continued)



```
LESSON1                           Text Page                                  1
______________________________________________________________________________
    This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.







______________________________________________________________________________
          Enter Text           Ctrl<E> to End          Ctrl<Q> to Quit
```

Input of a typical text page. Word wrap is automatic and current line can be corrected by backspacing. <CTRL><Q> erases entire page and returns to page number prompt. <CTRL><E> ends page input.

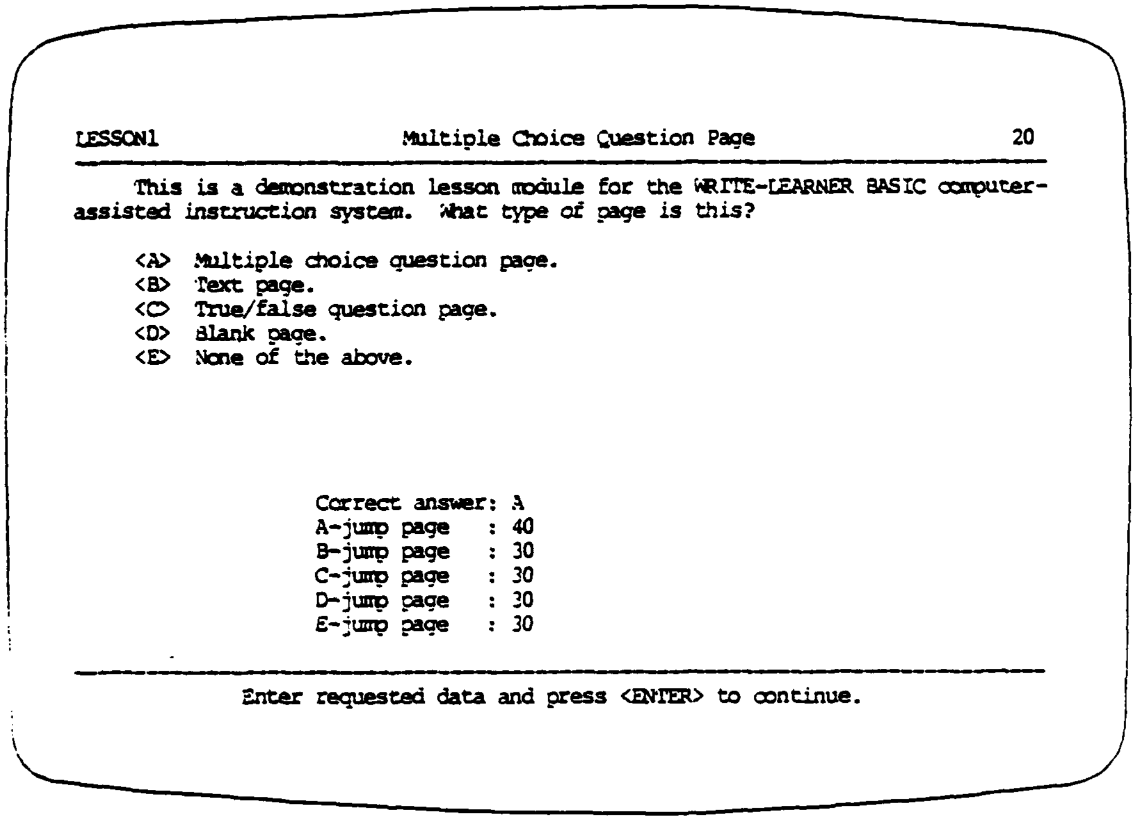
Figure E.2: Creating a Lesson File with WRITE/BAS (continued)



```
LESSON1                        Text Page                          1
-------------------------------------------------------------------
     This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.








                Jump-to page:
-------------------------------------------------------------------
            Enter requested data and press <ENTER> to continue.
```

Input of jump-to page following <CTRL><E> input during text entry. Jump-to page number is validated for being within legal limits.

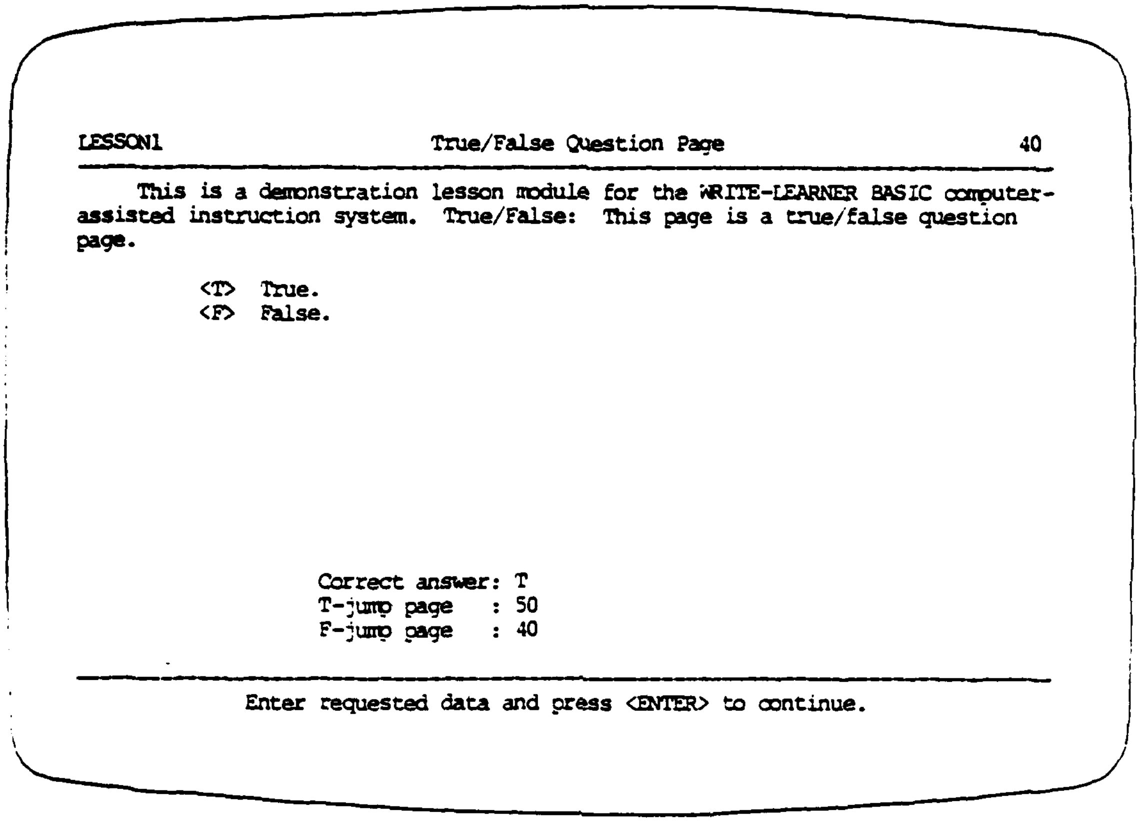
Figure E.2: Creating a Lesson File with WRITE/BAS (continued)



```
LESSON1                 Multiple Choice Question Page                   20
-----------------------------------------------------------------------------
     This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.  What type of page is this?

     <A>  Multiple choice question page.
     <B>  Text page.
     <C>  True/false question page.
     <D>  Blank page.
     <E>  None of the above.



                    Correct answer: A
                    A-jump page   : 40
                    B-jump page   : 30
                    C-jump page   : 30
                    D-jump page   : 30
                    E-jump page   : 30
-----------------------------------------------------------------------------
              Enter requested data and press <ENTER> to continue.
```

Input of correct answer and jump-to pages following <CTRL><E> input during multiple-choice question screen text entry. Correct answer is single letter key (A-E) input.

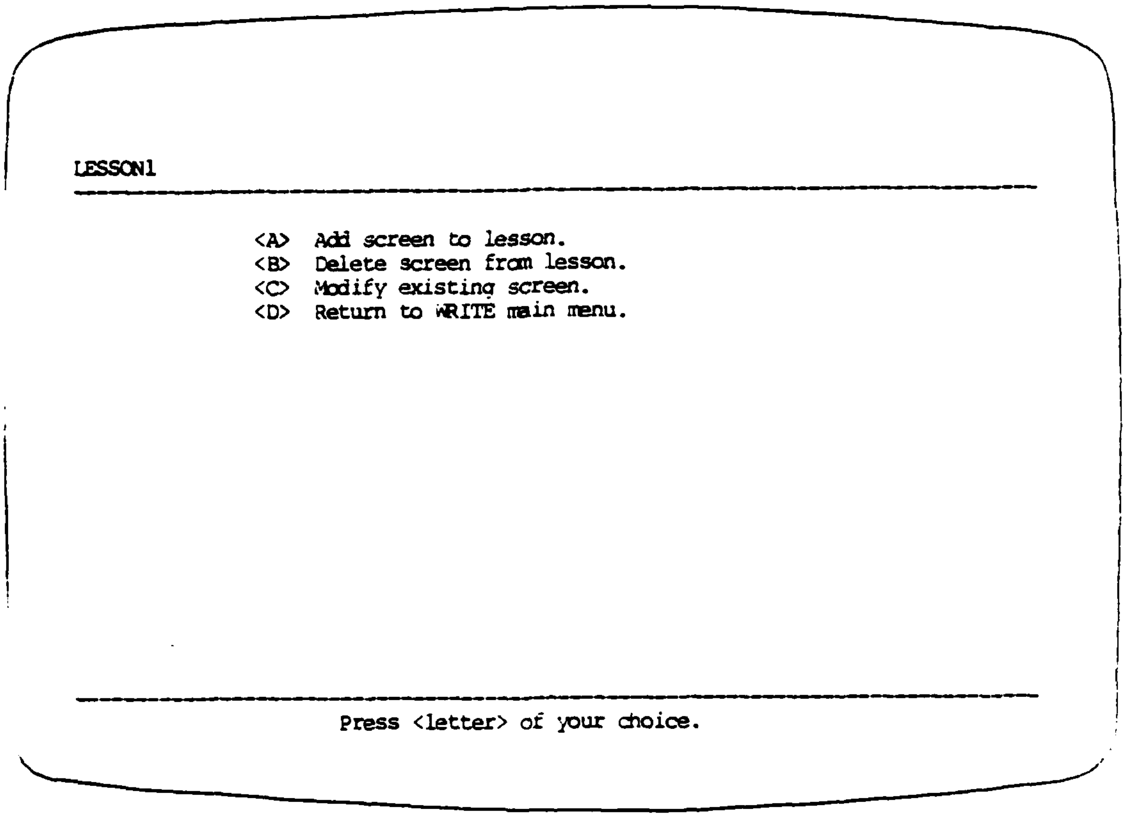
Figure E.2: Creating a Lesson File with WRITE/BAS (continued)



```
LESSON1                   True/False Question Page                   40
__________________________________________________________________________
     This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.  True/False:  This page is a true/false question
page.

          <T>  True.
          <F>  False.






                    Correct answer: T
                    T-jump page   : 50
                    F-jump page   : 40
__________________________________________________________________________
              Enter requested data and press <ENTER> to continue.
```

Input of correct answer and jump-to pages following <CTRL><E> input during true/false question screen text entry. Correct answer is single letter key (T/F) input.

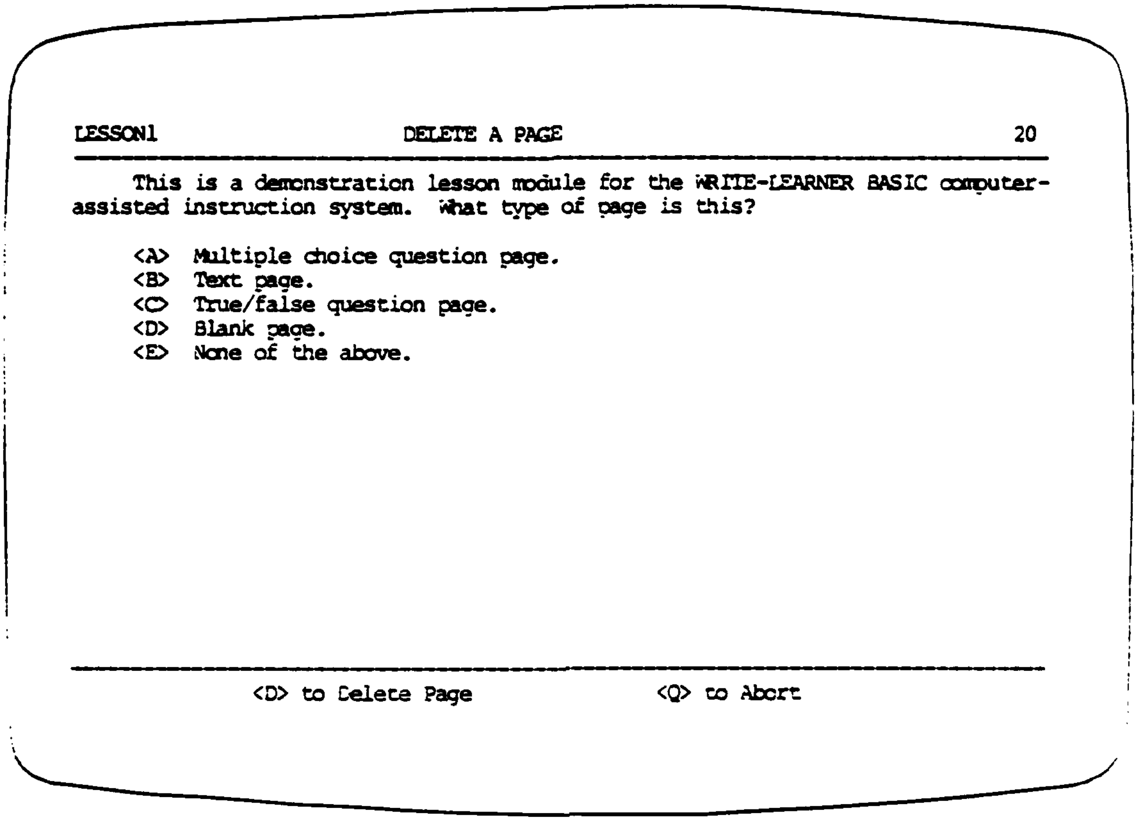
Figure E.2: Creating a Lesson File with WRITE/BAS (continued)



```
LESSON1
_____________________________________________________________

                <A>  Add screen to lesson.
                <B>  Delete screen from lesson.
                <C>  Modify existing screen.
                <D>  Return to WRITE main menu.

_____________________________________________________________
                       Press <letter> of your choice.
```

WRITE/BAS edit menu and input of desired function.

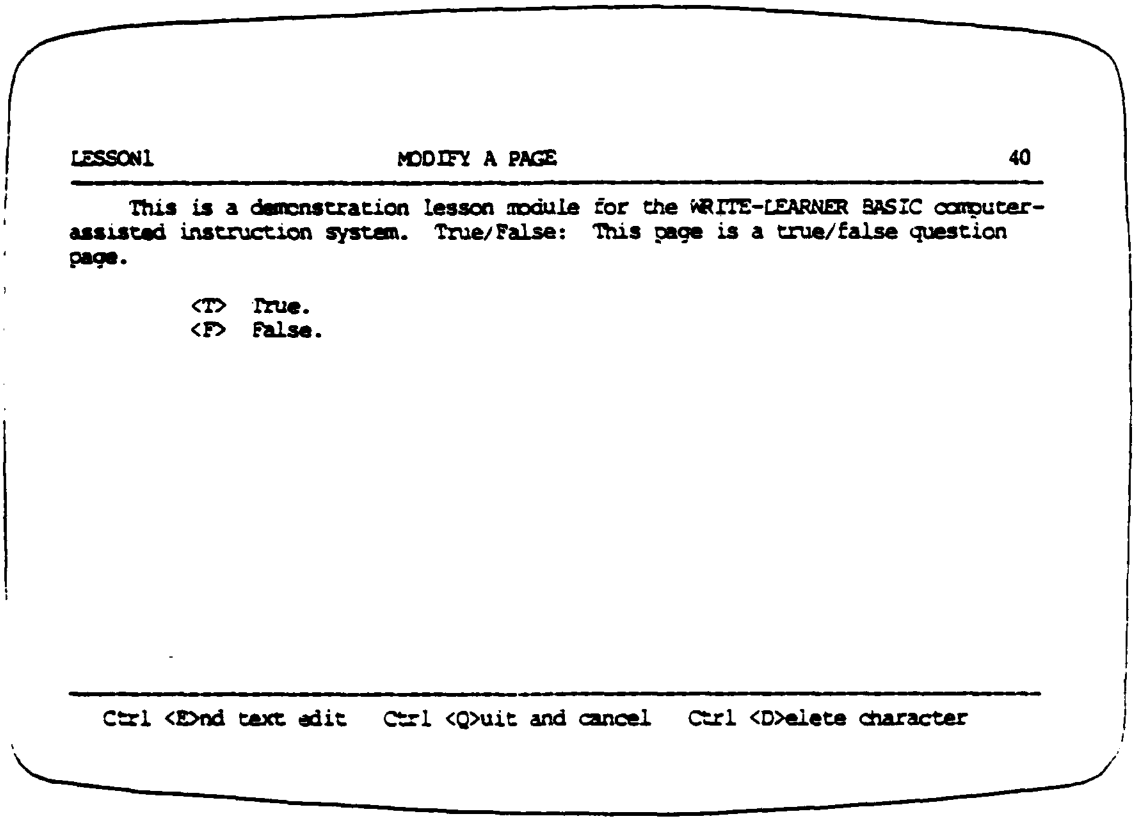
Figure E.3: Editing a Lesson File with WRITE/BAS



```
LESSON1                     DELETE A PAGE                          20
______________________________________________________________________
     This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.  What type of page is this?

     <A>  Multiple choice question page.
     <B>  Text page.
     <C>  True/false question page.
     <D>  Blank page.
     <E>  None of the above.




______________________________________________________________________
              <D> to Delete Page              <Q> to Abort
```

Display of page to delete following page number input and input of delete decision.

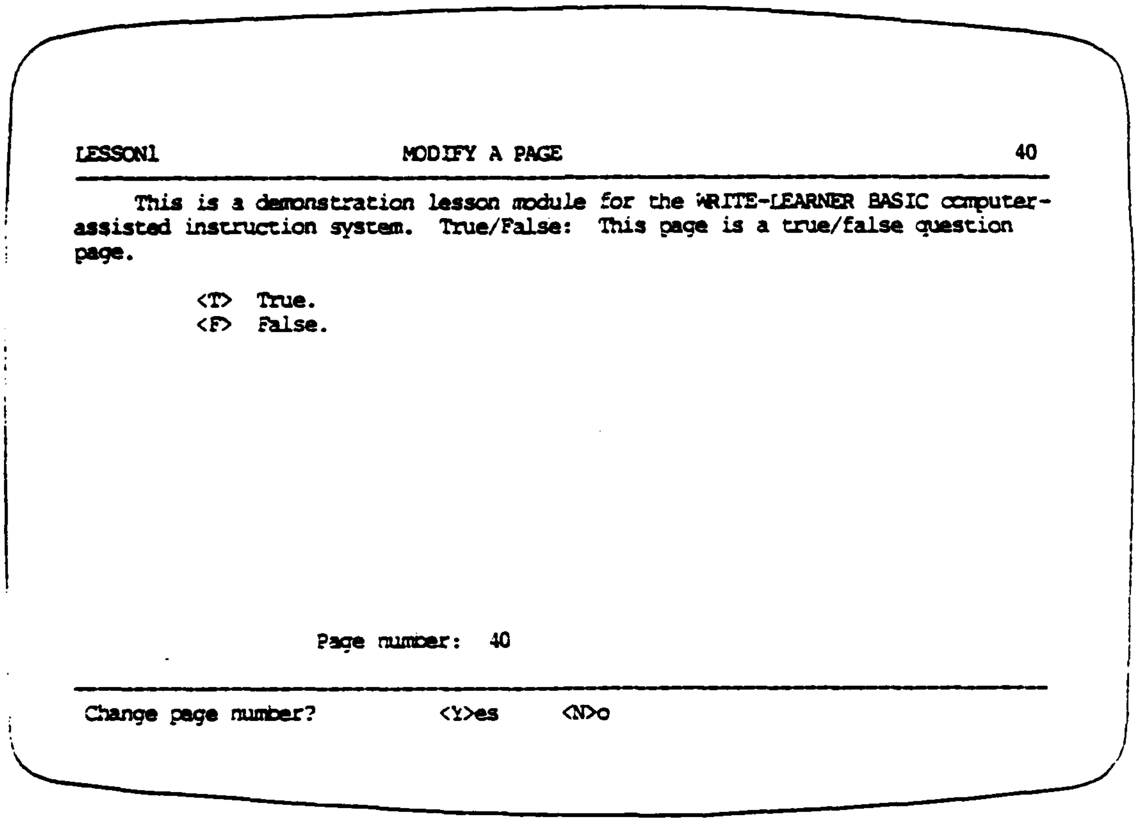
Figure E.3: Editing a Lesson File with WRITE/BAS (continued)



```
LESSON1                    MODIFY A PAGE                          40
--------------------------------------------------------------------
     This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.  True/False:  This page is a true/false question
page.

          <T>  True.
          <F>  False.

--------------------------------------------------------------------
  Ctrl <E>nd text edit   Ctrl <Q>uit and cancel   Ctrl <D>elete character
```

Display of page to modify following page number input. Arrow keys move cursor within limits of current page. Overstriking or deleting of characters is allowed.

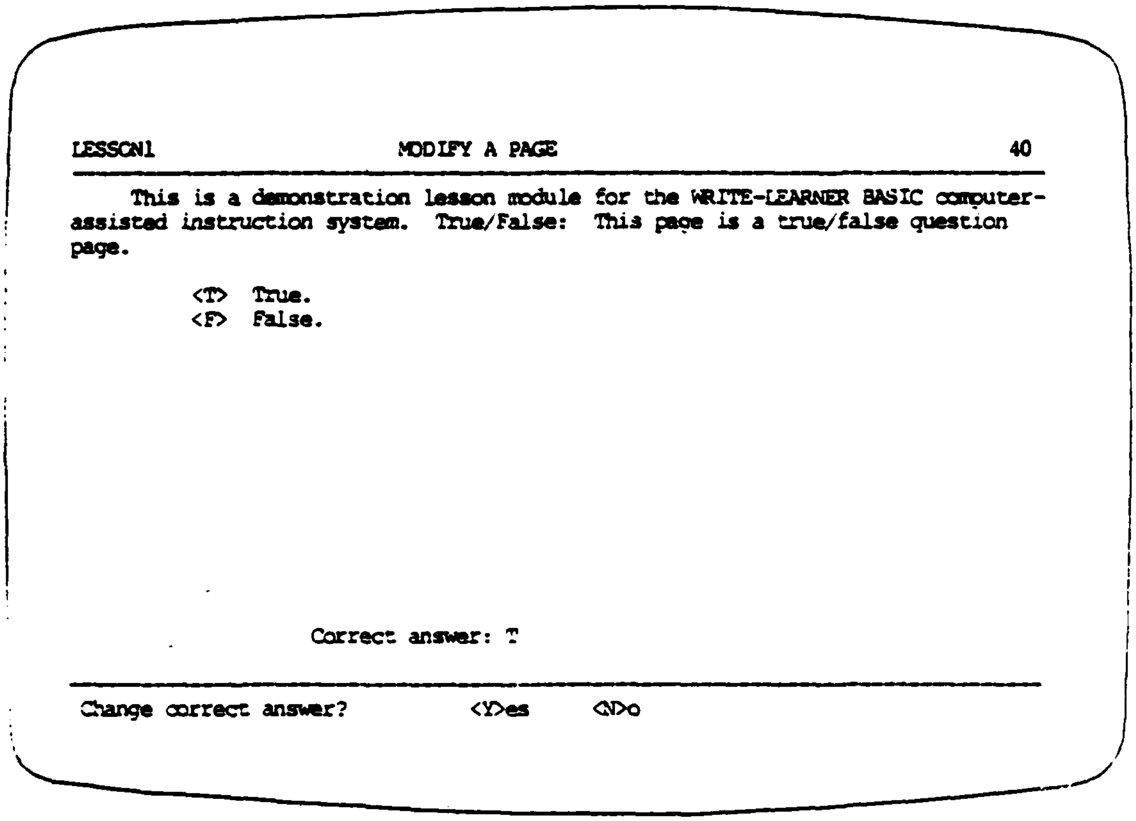
Figure E.3: Editing a Lesson File with WRITE/BAS (continued)



```
LESSON1                    MODIFY A PAGE                              40

    This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.  True/False:  This page is a true/false question
page.

           <T>  True.
           <F>  False.








                     Page number:  40

 Change page number?           <Y>es     <N>o
```

Changing lesson screen page number following
<CTRL><E> input during text editing.

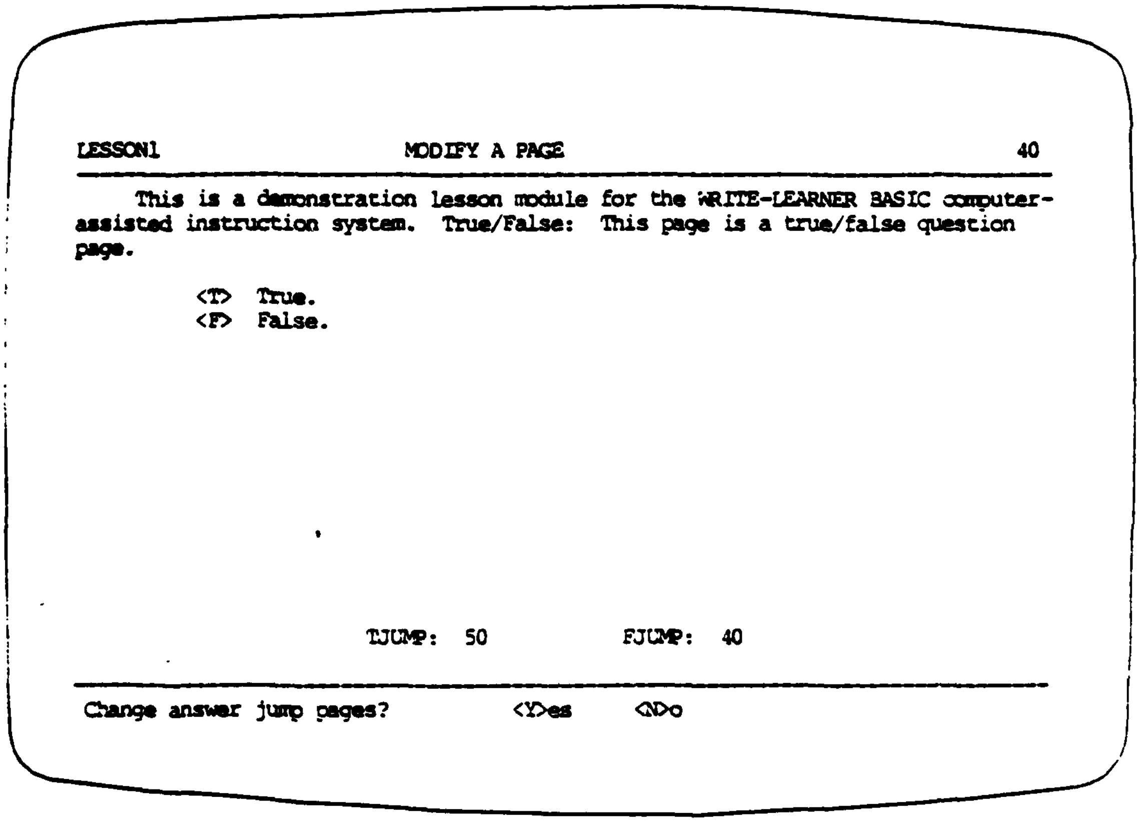
Figure E.3: Editing a Lesson File with WRITE/BAS (continued)



```
LESSON1                         MODIFY A PAGE                            40
____________________________________________________________________________
     This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.  True/False:  This page is a true/false question
page.

          <T>  True.
          <F>  False.










                    Correct answer: T
____________________________________________________________________________
Change correct answer?          <Y>es      <N>o
```

Changing correct answer for multiple-choice or true/false question page.

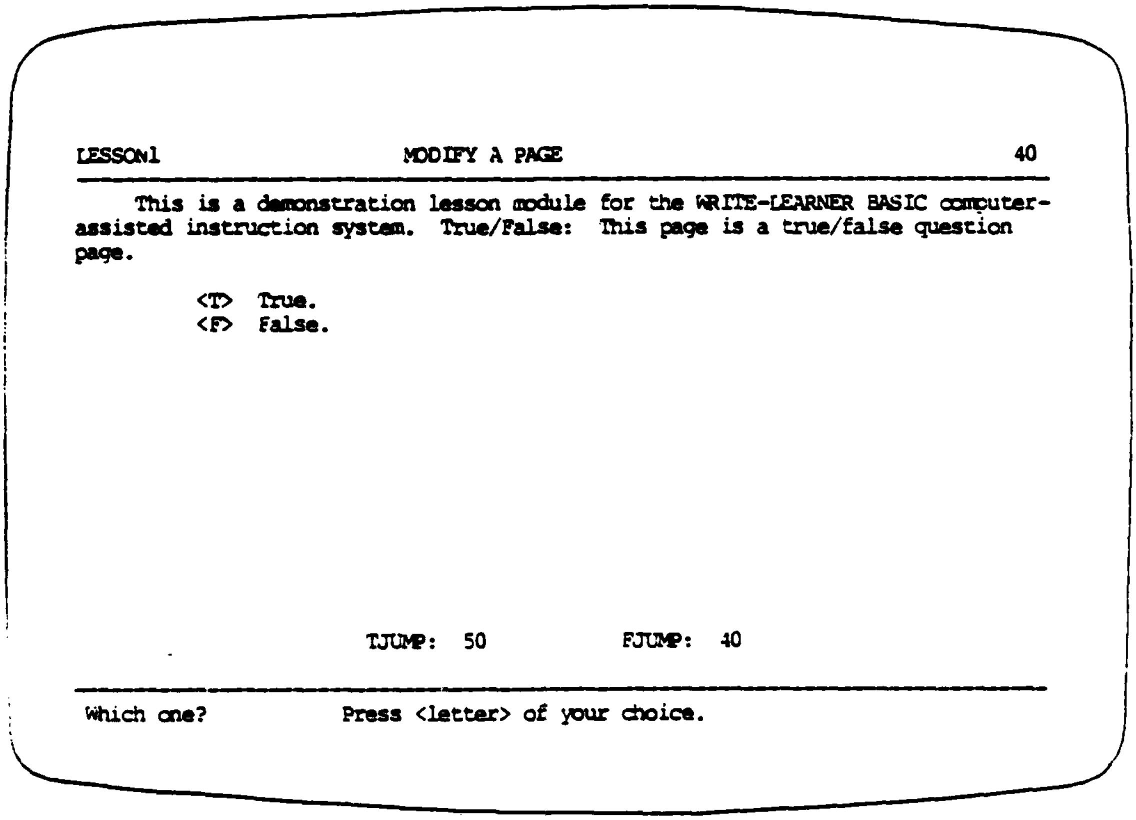
Figure E.3: Editing a Lesson File with WRITE/BAS (continued)



```
LESSON1                    MODIFY A PAGE                              40
________________________________________________________________________
     This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.  True/False:  This page is a true/false question
page.

          <T>  True.
          <F>  False.










                      TJUMP:  50            FJUMP:  40
________________________________________________________________________
Change answer jump pages?          <Y>es     <N>o
```

Changing jump-to page numbers.

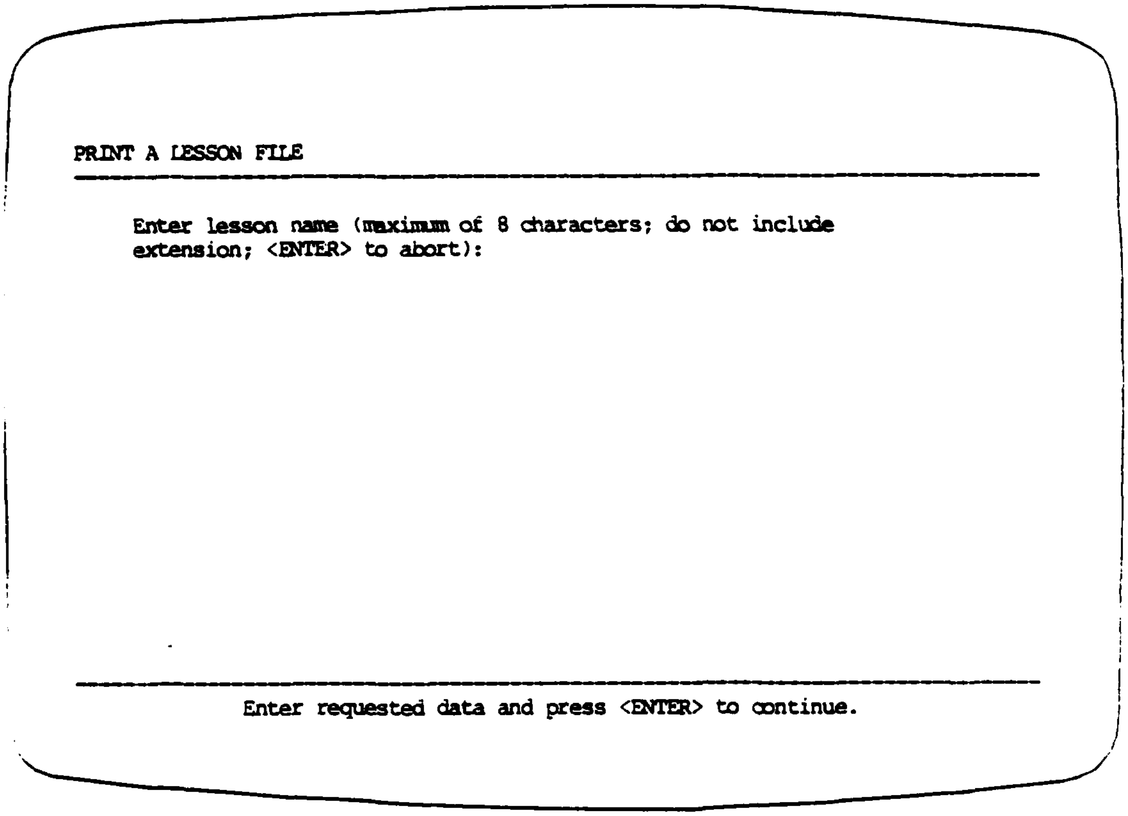
Figure E.3: Editing a Lesson File with WRITE/BAS (continued)



```
LESSON1                     MODIFY A PAGE                              40
---------------------------------------------------------------------------
     This is a demonstration lesson module for the WRITE-LEARNER BASIC computer-
assisted instruction system.  True/False:  This page is a true/false question
page.

          <T>  True.
          <F>  False.







                         TJUMP:  50           FJUMP:  40
---------------------------------------------------------------------------
Which one?           Press <letter> of your choice.
```

Changing jump-to page numbers.

Figure E.3: Editing a Lesson File with WRITE/BAS (continued)



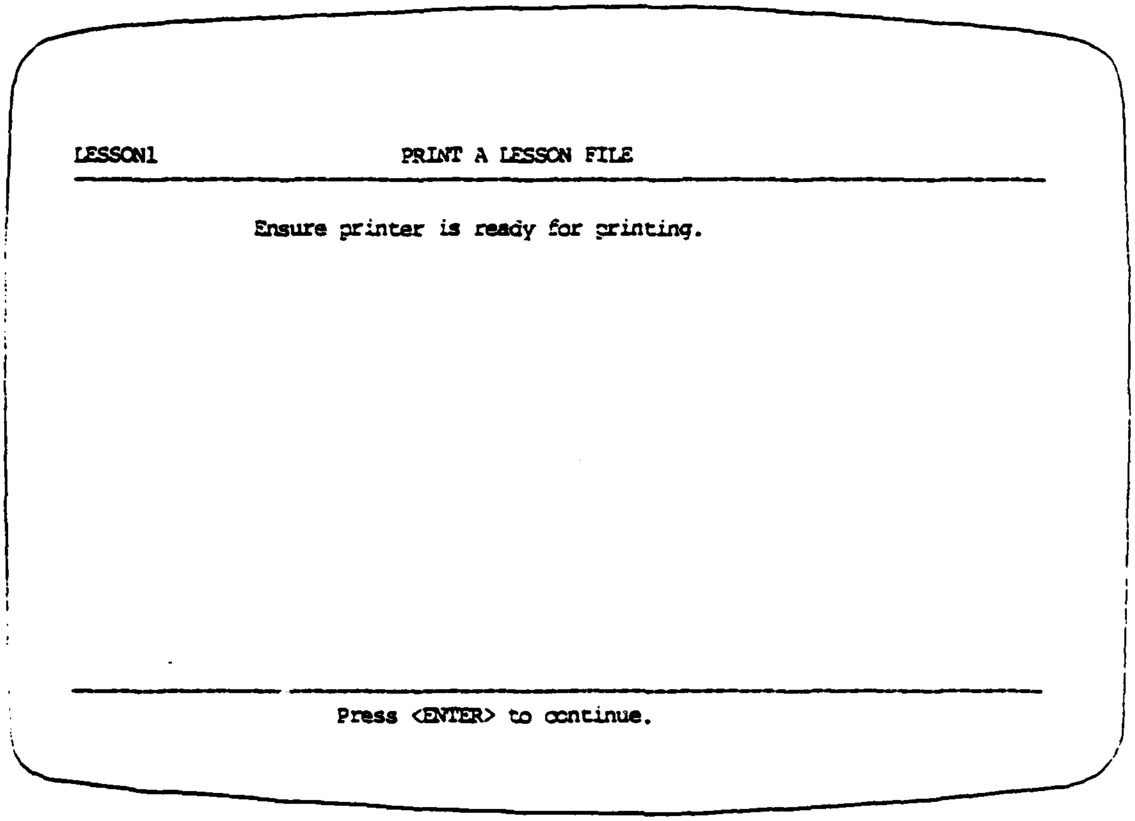
PRINT A LESSON FILE

Enter lesson name (maximum of 8 characters; do not include
extension; <ENTER> to abort):

Enter requested data and press <ENTER> to continue.

Input of lesson name to print. The screen for printing a student file report is similiar.

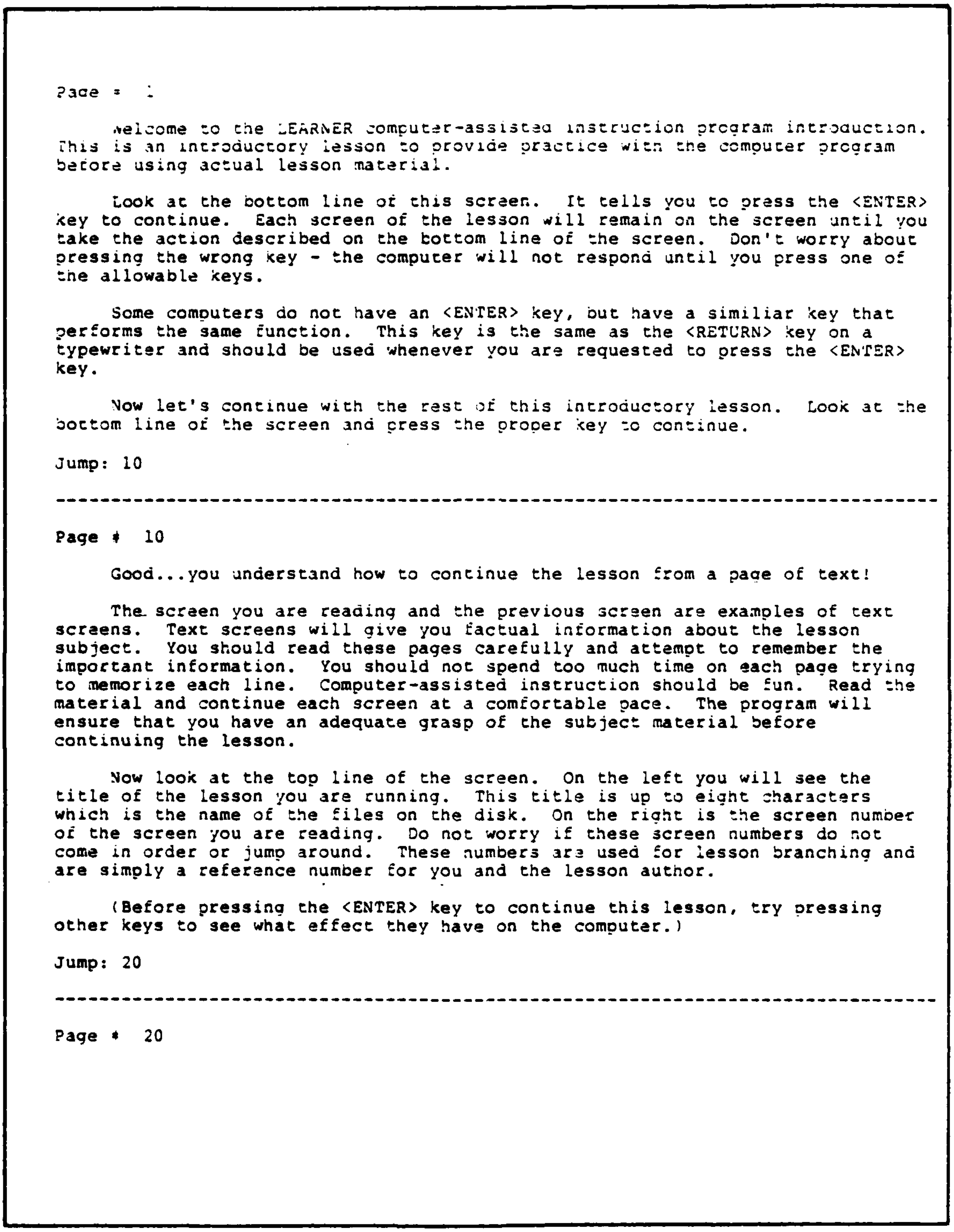
Figure E.4: Printing a Lesson File with WRITE/BAS



```
LESSON1                    PRINT A LESSON FILE

              Ensure printer is ready for printing.




                      Press <ENTER> to continue.
```

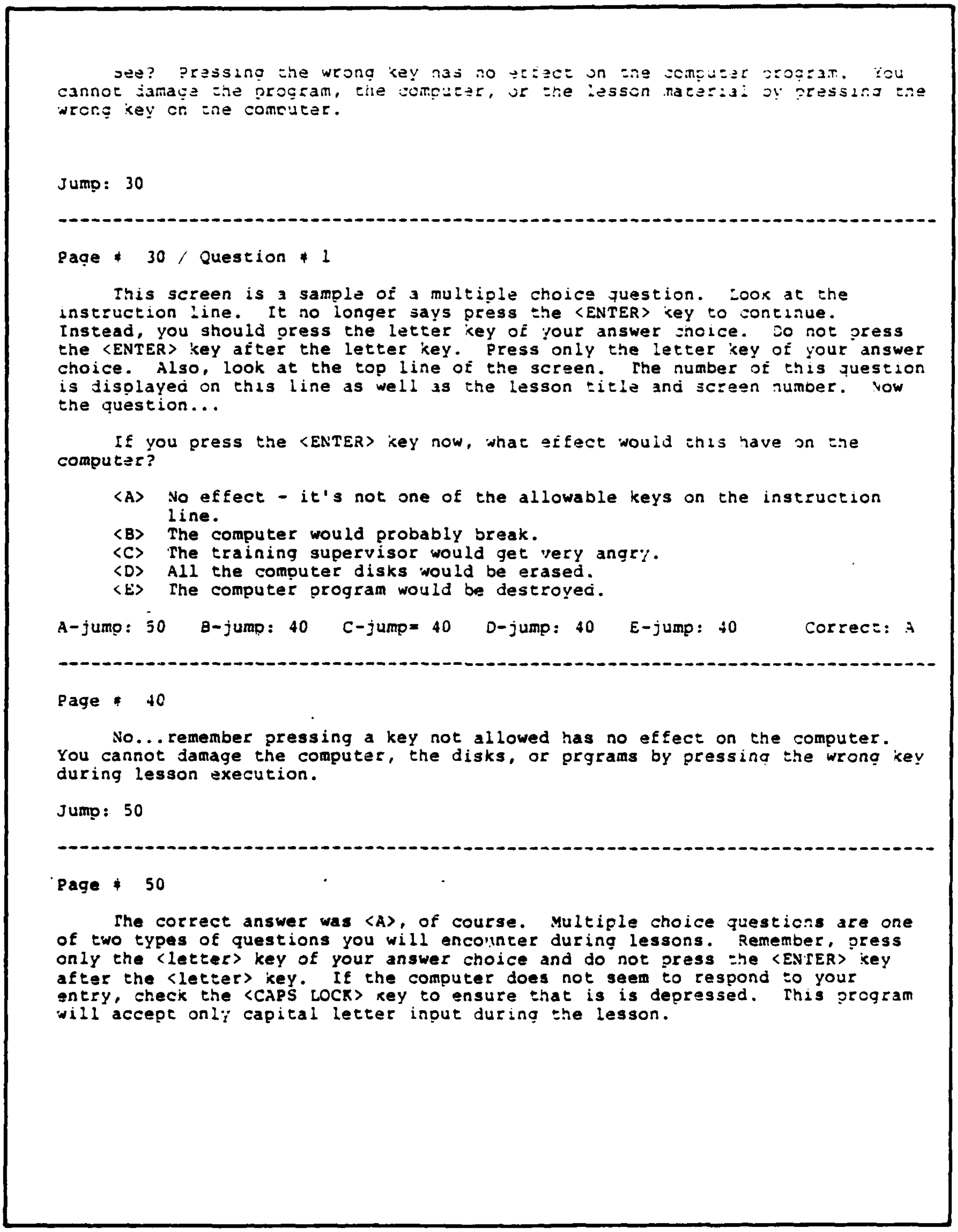
Printer instructions for printing a lesson file. The screen for printing a student file report is similiar.

Figure E.4: Printing a Lesson File with WRITE/BAS (continued)



```
Page # 1

     Welcome to the LEARNER computer-assisted instruction program introduction.
This is an introductory lesson to provide practice with the computer program
before using actual lesson material.

     Look at the bottom line of this screen.  It tells you to press the <ENTER>
key to continue.  Each screen of the lesson will remain on the screen until you
take the action described on the bottom line of the screen.  Don't worry about
pressing the wrong key - the computer will not respond until you press one of
the allowable keys.

     Some computers do not have an <ENTER> key, but have a similiar key that
performs the same function.  This key is the same as the <RETURN> key on a
typewriter and should be used whenever you are requested to press the <ENTER>
key.

     Now let's continue with the rest of this introductory lesson.  Look at the
bottom line of the screen and press the proper key to continue.

Jump: 10

--------------------------------------------------------------------------------

Page # 10

     Good...you understand how to continue the lesson from a page of text!

     The screen you are reading and the previous screen are examples of text
screens.  Text screens will give you factual information about the lesson
subject.  You should read these pages carefully and attempt to remember the
important information.  You should not spend too much time on each page trying
to memorize each line.  Computer-assisted instruction should be fun.  Read the
material and continue each screen at a comfortable pace.  The program will
ensure that you have an adequate grasp of the subject material before
continuing the lesson.

     Now look at the top line of the screen.  On the left you will see the
title of the lesson you are running.  This title is up to eight characters
which is the name of the files on the disk.  On the right is the screen number
of the screen you are reading.  Do not worry if these screen numbers do not
come in order or jump around.  These numbers are used for lesson branching and
are simply a reference number for you and the lesson author.

     (Before pressing the <ENTER> key to continue this lesson, try pressing
other keys to see what effect they have on the computer.)

Jump: 20

--------------------------------------------------------------------------------

Page # 20
```

Figure E.5: Sample of Printed Lesson File



See? Pressing the wrong key has no effect on the computer program. You cannot damage the program, the computer, or the lesson material by pressing the wrong key on the computer.

Jump: 30

Page # 30 / Question # 1

This screen is a sample of a multiple choice question. Look at the instruction line. It no longer says press the <ENTER> key to continue. Instead, you should press the letter key of your answer choice. Do not press the <ENTER> key after the letter key. Press only the letter key of your answer choice. Also, look at the top line of the screen. The number of this question is displayed on this line as well as the lesson title and screen number. Now the question...

If you press the <ENTER> key now, what effect would this have on the computer?

<A> No effect - it's not one of the allowable keys on the instruction line.
<B> The computer would probably break.
<C> The training supervisor would get very angry.
<D> All the computer disks would be erased.
<E> The computer program would be destroyed.

A-jump: 50   B-jump: 40   C-jump= 40   D-jump: 40   E-jump: 40   Correct: A

Page # 40

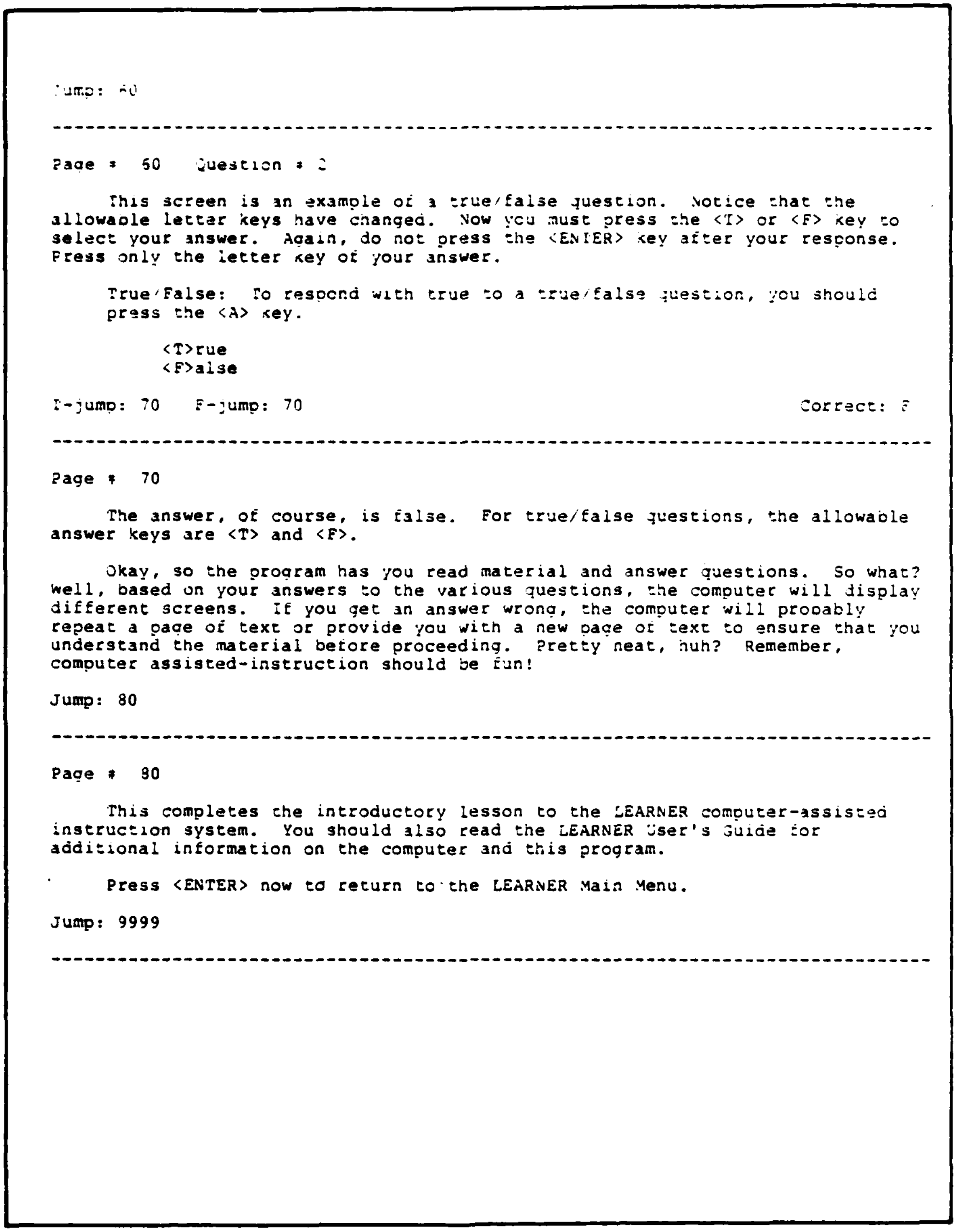
No...remember pressing a key not allowed has no effect on the computer. You cannot damage the computer, the disks, or prgrams by pressing the wrong key during lesson execution.

Jump: 50

Page # 50

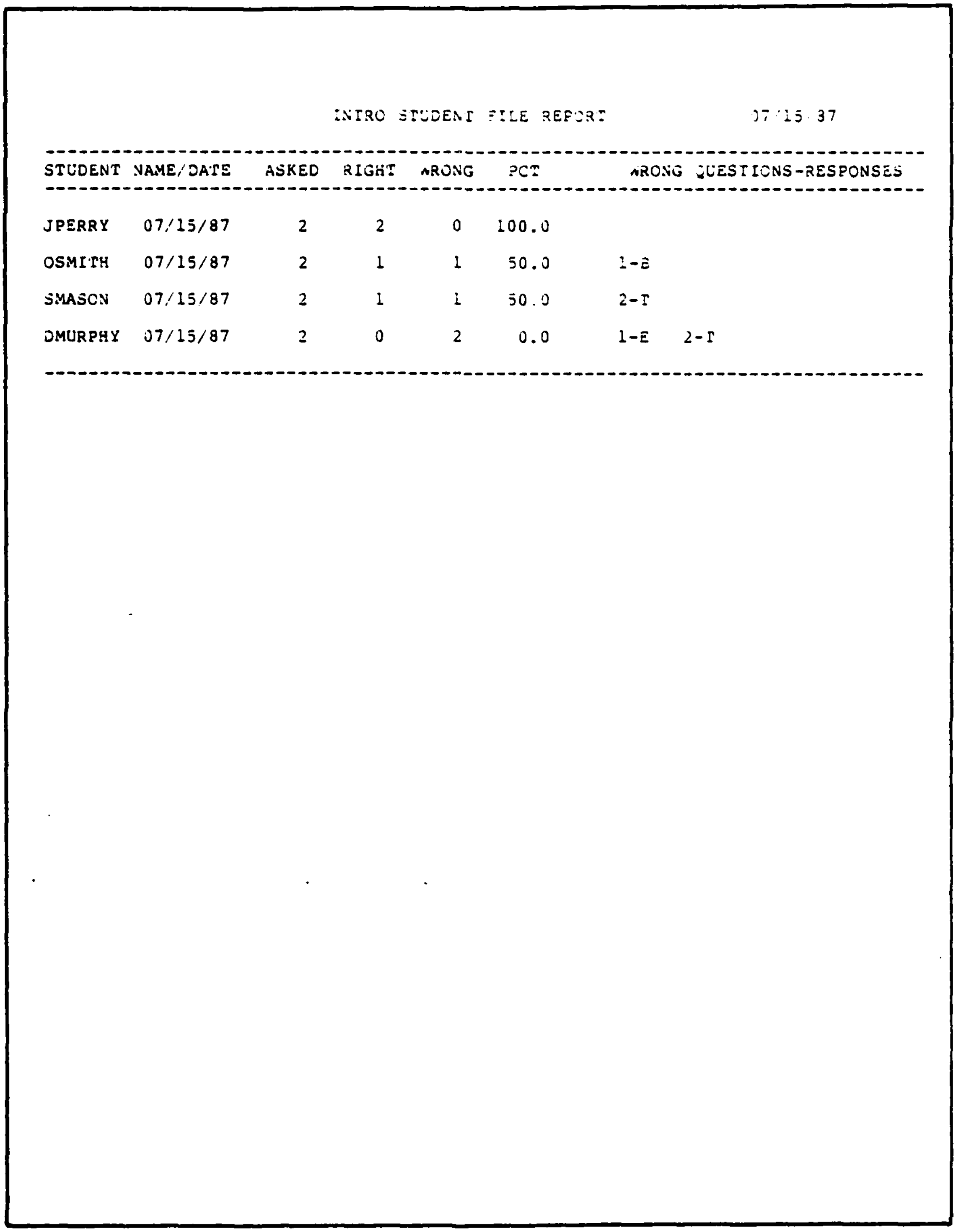
The correct answer was <A>, of course. Multiple choice questions are one of two types of questions you will encounter during lessons. Remember, press only the <letter> key of your answer choice and do not press the <ENTER> key after the <letter> key. If the computer does not seem to respond to your entry, check the <CAPS LOCK> key to ensure that is is depressed. This program will accept only capital letter input during the lesson.

Figure E.5: Sample of Printed Lesson File (continued)



```
Jump: 60

---------------------------------------------------------------------------------

Page # 60   Question # 2

     This screen is an example of a true/false question.  Notice that the
allowable letter keys have changed.  Now you must press the <T> or <F> key to
select your answer.  Again, do not press the <ENTER> key after your response.
Press only the letter key of your answer.

     True/False:  To respond with true to a true/false question, you should
     press the <A> key.

          <T>rue
          <F>alse

T-jump: 70   F-jump: 70                                            Correct: F

---------------------------------------------------------------------------------

Page # 70

     The answer, of course, is false.  For true/false questions, the allowable
answer keys are <T> and <F>.

     Okay, so the program has you read material and answer questions.  So what?
Well, based on your answers to the various questions, the computer will display
different screens.  If you get an answer wrong, the computer will probably
repeat a page of text or provide you with a new page of text to ensure that you
understand the material before proceeding.  Pretty neat, huh?  Remember,
computer assisted-instruction should be fun!

Jump: 80

---------------------------------------------------------------------------------

Page # 80

     This completes the introductory lesson to the LEARNER computer-assisted
instruction system.  You should also read the LEARNER User's Guide for
additional information on the computer and this program.

     Press <ENTER> now to return to the LEARNER Main Menu.

Jump: 9999

---------------------------------------------------------------------------------
```

Figure E.5: Sample of Printed Lesson File (continued)



```
                       INTRO STUDENT FILE REPORT              07/15/87
-------------------------------------------------------------------------------
STUDENT NAME/DATE   ASKED  RIGHT  WRONG   PCT        WRONG QUESTIONS-RESPONSES
-------------------------------------------------------------------------------

JPERRY   07/15/87     2      2      0   100.0

OSMITH   07/15/87     2      1      1    50.0      1-B

SMASON   07/15/87     2      1      1    50.0      2-T

DMURPHY  07/15/87     2      0      2     0.0      1-B   2-T

-------------------------------------------------------------------------------
```

Figure E.6: Sample of Printed Student File Report

Appendix F: WRITE/BAS User's Guide

Contents

Introduction

WRITE/BAS is a program written in the BASIC programming language which is one component of the WRITE-LEARNER computer-assisted instruction system. WRITE will enable you to create interactive, computer-assisted instruction courseware to be administered with the LEARNER/BAS program. This user's guide is intended to provide the knowledge required to use the WRITE program with a minimum of computer knowledge.

The courseware you develop can consist of up to 200 lesson screens consisting of three different types of pages or screens. Text pages contain up to twenty lines of text to present information to the student. Multiple-choice question pages consist of a question and five (letters A-E) allowable question responses. True/false question pages consist of a statement with two allowable question responses (T/F). Each page also has an appropriate number of jump pages to permit variable lesson branching. For example, an incorrect response to a multiple choice question page may branch to a series of text pages containing remedial or amplified information about the subject area while a correct response skips this supplemental information. You should note that a "lesson" can consist only of question pages and can therefore be used as a test generator.

Question responses must be entered as text lines on the screen. Responses to multiple choice questions must be entered as "<A> answer response". The allowable responses to true/false questions should be entered as "<T>rue/<F>alse". The "< >" symbol (indicating a keyboard key) around the allowable responses is to maintain consistency with the remainder of the program. You should run the introductory lesson of LEARNER/BAS for formats of text, multiple-choice question, and true/false question screens.

Notes for Courseware Administrators

This user's guide was developed primarily for use with a Tandy-Radio Shack Model IV or Zenith Z-248 microcomputer system. With the wide diversity of MS-DOS and TRSDOS computers available and the diversity of operating systems and disk-operating system "shells" in use, an "all-encompassing" user's guide for this system is not practible. This guide is being copied (as written) in ASCII format on the WRITE-LEARNER distribution disk. Courseware administrators will use this guide to develop local instructions depending on computers available and operating systems in use. Also, the use of job control language files (Model IV) or auto-batch command files (MS-DOS) to start the program is encouraged.

The user may require some BASIC programming knowledge depending on local computer configuration. The WRITE program is configured to operate with the system disk on a disk drive designated drive 1 (TRS-DOS) or A (MS-DOS). If the local computer configuration does not permit this, lines 1550, 1555, and 1560 will require modification to designate the proper drive.

Another problem may exist if numerous pages are deleted from a lesson file. WRITE/BAS does not delete records from the lesson text file but rather fills the records with "*" characters. This may lead to excessively long text files. Due to the large numbers of computer configurations available, deleting these records and collapsing the text file is not accomplished by this program. Consult a BASIC programmer to develop a utility to accomplish this task. In general, the procedure is to read a number of records from the text file into memory and then write the records not flagged for deletion to a new text file. Following this procedure, a new lesson table file must be created by entering the WRITE/BAS lesson editing module and immediately selecting the exit editor option. This procedure forces recreation of the lesson table file.

In the event that a lesson text file exists on a disk without the corresponding lesson table file (due to data loss or erroneous disk copying) the WRITE/BAS editing module will return an error message. To recover from this error, copy any lesson table file to the disk (such as the LEARNER/BAS introduction lesson table file, INTRO/TAB) to the required disk. Rename the file from "INTRO/TAB" to "<Lesson Filename>/TAB". Enter the WRITE/BAS editing module and immediately select the exit editor option. This procedure forces recreation of the lesson table file. The lesson text file can then be edited or used in a normal manner.

Getting Started

WRITE/BAS operates under the BASIC programming language using data disks from the WRITE-LEARNER package and locally developed courseware data disks. To start the WRITE program, take the following actions:

Step 1 Turn on the computer and monitor.

Step 2 Load BASIC into the computer. On AFIT Z-248's, this is accomplished by pressing the <B> key while at the main menu display. On the Model IV,type "BASIC <ENTER>" at the DOS ready prompt.

Step 3 Insert the WRITE-LEARNER system disk in drive A (Z-248) or drive 1 (Model IV).

Step 4 Depending on computer configuration, type one of the following line exactly as it appears then press the <ENTER> key:

```
RUN "A:WRITE.BAS" (Z-248)
RUN "WRITE/BAS:1" (Model IV)
```

Step 5 The WRITE main menu will be displayed at this point. Select the option of your choice and press the corresponding <letter> key. Do not press the <ENTER> key after your selection; press only the <letter> key.

Step 6 Note that option <E> on the WRITE main menu ends the program. This is the normal termination procedure for this program. Pressing the <E> key will return the computer to the DOS ready prompt.

Creating a Lesson File

To create a new lesson file, take the following actions:

Step 1 Select <A> at the WRITE main menu.

Step 2 Insert a properly formatted (recommend a newly formatted disk) in the disk drive when directed.

Step 3 Enter the lesson filename when directed.

Step 4 Enter the page number when directed. Page numbers must be between 1 and 9998. The first page of a lesson must be page 1! The program will not allow you to duplicate page numbers.

Step 5 Select the type of page to enter and press the corresponding <letter> key.

Step 6 The lesson name, page type, and page number will be displayed on the top line of the screen. You are now in the screen entry mode. Enter up to 20 lines of text as you want it to appear in the lesson. In this mode, word wrap is automatic (i.e., when you continue past the end of a line, the line will be broken at the last space, dash, or slash and the remainder of the line moved to the next text line. The following special keys are available in the screen entry mode:

<BACKSPACE> or <left arrow>: Permits correction of the current line only. Backspaces the cursor and erases any text under the cursor.

<TAB> or <right arrow>: Tabs the cursor 5 spaces; does not function if insufficient room remains on the current line to tab.

<ENTER>: Terminates entry of the current line and moves the cursor to the next line.

<CTRL> and <E> (pressed together): Terminates entry of the current page and requests additional information about the page.

<CTRL> and <Q> (pressed together): Quits entry of the current page without saving! Use to totally erase a page and start over.

Step 7 After pressing <CTRL><E> to end page input, additional data will be requested on the lower 5 lines of the text page. If you have used the full page, some of these text lines will be erased from the screen but are still in the lesson. The lines are erased only to provide room to enter the additional data. Enter the additional data in accordance with the prompts. For correct answer prompts, press only the <letter> key of the correct response (i.e., A-E for multiple choice questions or T/F for true/false questions). Jump pages are the corresponding page of the lesson that will be displayed next depending on the student's answer. Jump pages must be between 1 and 9999 and need not have been already entered to use. A jump page of 9999 will terminate lesson execution when run in the LEARNER program.

Step 8 To terminate lesson input, enter a page number of 9999 when requested. This action will terminate the create lesson module of the program, generate the required files on the disk and return to the WRITE main menu.

Editing a Lesson File

To edit an existing lesson file, take the following actions:

NOTE: To edit a lesson file, the text file and table file must be on the same disk! If you copy a lesson file from one disk to another, ensure that both files are copied!

Step 1 Select <B> at the WRITE main menu.

Step 2 Insert the disk containing the lesson files in the disk drive when directed.

Step 3 Enter the lesson filename when directed.

Step 4 Select the edit action desired and press the corresponding <letter> key.

Step 5 Selecting <A> to add page(s) to the lesson enters the same module as creating a lesson file. Add pages to the lesson in accordance with the above instructions. To exit adding pages to the lesson, enter a page number of 9999 when requested. This action will return to the edit menu.

Step 6 Selecting <B> to delete a page will display the selected page to ensure that you want to delete that page. Press <D> to delete the page or <Q> to return to the edit menu without deleting the page.

Step 7 Selecting <C> to modify a page will display the selected page and enter a full screen editing mode. This is a limited full screen editor intended to make minor corrections to an existing page. Lines cannot be added to the screen! If major modifications are required, delete the page using the <B> option the add the same numbered page using the <A> option. The following special keys are available in the screen edit mode:

<arrow> keys: Move the cursor around the screen within the confines of the existing page (i.e., you cannot move down past the current last line).

<CTRL> and <D> (pressed together) or <BACKSPACE> or <DELETE>: Deletes the character under the cursor.

<valid ASCII> keys: Replaces the character under the cursor with the input key.

<CTRL> and <Q> (pressed together): Quits the editor without saving any changes to the page.

<CTRL> and <E> (pressed together): Ends text editing and displays additional information to be changed. Answer the prompts by pressing the appropriate <letter> key.

Step 8 Selecting <D> ends lesson editing, updates appropriate files on the disk, and returns to the WRITE main menu.

Printing Lesson Files and Student File Reports

Lesson files and student files can be printed in a formatted manner on any available printer. Select these options at the WRITE main menu and follow the computer prompts to print these files.

Lesson File Problems

This checklist is provided for checking a lesson file printed copy. Use this checklist to verify lesson operation or to identify problems.

Step 1 A page numbered 1 must exist in each lesson.

Step 2 A text page cannot jump to itself and all answer jumps of a question cannot jump to itself (i.e., never-ending loop created).

Step 3 A valid numbered page must exist for all jump pages except 9999.

Step 4 All branches through a lesson must end with a jump to page 9999 (terminates lesson execution). Recommend that the last page of a lesson be a text page with a jump page of 9999.

Error Messages

As with any computer program, every precaution has been taken to ensure that the program is error free; however, unforeseen errors may occur. In the event that an error occurs, an error message will appear on the screen. Pressing <ENTER> will close all open files and return to the WRITE main menu. You should select the edit option and immediately end editing (option <D>) to force recreation of the lesson table file. Data may be lost depending on the error. Refer to the appropriate BASIC reference for translation of error codes.

Bibliography

1. Bahniuk, Dr. Margaret. "The Value in Computer-Based Training," Office Administration and Automation, 45 (8): 85 (August 1984).

2. Bork, Alfred. Personal Computers for Education. New York NY: Harper & Row, Publishers, Inc., 1985.

3. Boyd, Warren A. and John R. Eldridge. "Beyond User Friendly," Training and Development Journal, 38 (12): 36-8 (December 1984).

4. Burlage, John. "Carrier America Keeps Parts Problems in Check," Navy Times, 36 (4): 4 (November 10, 1986).

5. -----. "Navy Plans Longer Sea Tours for Some AK's," Navy Times, 36 (2): 2 (October 27, 1986).

6. -----. "AK Shortage Cited in Parts Tracking Ills," Navy Times, 36 (1): 1+ (October 20, 1986).

7. -----. "Audit Hits Supply Practices, But Praises Corrective Steps," Navy Times, 35 (52): 4 (October 13, 1986).

8. Fauley, Franz E. "The New Training Technologies: Their Rocky Road to Acceptance," Training and Development Journal, 37 (12): 22-5 (December 1983).

9. -----. "Computer Based Education," Training and Development Journal, 34 (11): 20-1 (November 1980).

10. General Accounting Office. Navy's Progress in Improving Physical Inventory Controls and the Magnitude, Causes, and Impact of Inventory Record Inaccuracies in the Army, Air Force, and Defense Logistics Agency. GAO/NSIAD-84-9. Gaithersburg MD, November 4, 1983.

11. Govaerts, Kenneth C. and Thomas A. Grillot. "Computer-Based Industrial Training: A Primer," Training and Development Journal, 38 (11): 28-31 (November 1984).

12. Hillelsohn, Michael J. "How to Think About CBT," Training and Development Journal, 38 (12): 42-4 (December 1984).

13. Kearsley, . and Michael J. Hillelsohn. "How and Why (And Why Not) We Use Computer-Based Training," Training and Development Journal, 38 (1): 21-2+ (January 1984).

14. Kindel, Stephen and Ellen Benoit. "Hello, Mr. Chip," Forbes, 133 (9): 132+ (April 23, 1984).

15. Naval Supply Systems Command. Naval Aviation Supply Officer Program. NAVSUP Publication 550. Washington DC, 30 November 1984.

16. Noel, Jr., John V. and Frank E. Bassett. Division Officer's Guide (Second Edition). Annapolis MD: Naval Institute Press, 1976.

17. "Over the Side," Time, 126 (5): 18 (August 5, 1985).

18. Radio Shack Disk System Owner's Manual, TRS-80, Model 4. Fort Worth TX: Tandy Corporation, 1983.

19. Reynolds, Angus. "An Introduction to Computer-Based Learning," Training and Development Journal, 37 (5): 34-8 (May 1983).

20. Reynolds, Angus and Richard Davis. "The Five Most Frequent Questions (Plus One) About Computer-Based Learning," Training and Development Journal, 37 (5): 42+ (May 1983).

21. Tonkin, Bruce. "Choose Your Weapon: Basic, Pascal, or C," 80Micro, 84: 96+ (January 1987).

22. Walker, Decker F. "Reflections on the Educational Potential and Limitations of Microcomputers," Introduction to Microcomputer Applications in Education, edited by Lonnie J. Echternacht and others. Columbia MO: Instructional Materials Laboratory, University of Missouri-Columbia, 1984.

VITA

Lieutenant Robert Mason was born on 15 July 1952 in Dallas, Texas. He graduated from high school in Richmond, Virginia in 1970 and enlisted in the U. S. Navy in November of that year. He was selected for the Navy Enlisted Scientific Education Program (NESEP) in 1974 and attended Auburn University from which he received the degree of Bachelor of Science in Aerospace Engineering in 1978. He was concurrently commissioned as a Naval Supply Officer. He subsequently served as Supply Officer of USS NATHAN HALE (SSBN-623)(GOLD); Aviation Support Officer at Naval Air Station, Key West, Florida; and Aviation Support Officer of USS AMERICA (CV-66). Lieutenant Mason holds warfare qualifications as Submarine Supply Officer and Naval Aviation Supply Officer. He entered the School of Systems and Logistics, Air Force Institute of Technology, in May 1986. He is married to the former Roberta Lynn Johnson of Richmond, Virginia, and has three daughters.

Permanent address: 721 Sunburst Lane
Dallas, Texas 75218

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

AD-A187 [illegible]

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release;
distribution unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
AFIT/GLM/LSR/87S-45

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Systems and
Logistics

6b. OFFICE SYMBOL *(If applicable)*
AFIT/LSM

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State, and ZIP Code)*
Air Force Institute of Technology
Wright-Patterson AFB OH 45433-6583

7b. ADDRESS *(City, State, and ZIP Code)*

8a. NAME OF FUNDING/SPONSORING ORGANIZATION

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State, and ZIP Code)*

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*
See Box 19

12. PERSONAL AUTHOR(S)
Robert Mason, B.S., LT, SC, USN

13a. TYPE OF REPORT
MS Thesis

13b. TIME COVERED
FROM ________ TO ______

14. DATE OF REPORT *(Year, Month, Day)*
1987 September

15. PAGE COUNT
234

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
05	12	
06	05	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Computer-assisted Instruction, Training,
Supply, Navy

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Title: A GENERAL APPLICATION COMPUTER-ASSISTED
INSTRUCTION SYSTEM FOR MICROCOMPUTERS

Thesis Chairman: John A. Stibravy, Major, USAF
Associate Professor of Technical Communications

Approved for publ
Lynn Wolaver
LYNN E. WOL
Dean for
Air Force
Wright-Patterson
24 Sept 87

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED □ SAME AS RPT. □ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
John A. Stibravy, Major, USAF

22b. TELEPHONE *(Include Area Code)*
(513) 255-6761

22c. OFFICE SYMBOL
AFIT/LSR

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

Block 19.

This study provides supply managers with an alternate method of augmenting Naval aviation storekeeper training to improve supply support management at operating sites. A microcomputer-based computer-assisted instruction system was developed which has much broader and significant application. The system can be used in any subject area to develop, administer, and monitor training and is applicable to all Navy, Air Force, Army, and other Department of Defense components. The potential for cost savings and improved operational capability through the use of this system is unlimited.

The system consists of two computer programs written in the BASIC programming language, program documentation, and a user's guide for the system. The system was developed on a Radio Shack, TRS-80, Model 4 microcomputer and converted to operate on a Zenith, Z-248, IBM-AT/PC compatible microcomputer.

The system creates and administers interactive computer-assisted instruction lessons consisting of up to 200 text, multiple choice question, and true/false question screens. Variable branching is allowed from question pages depending on student answer input.

LEARNER/BAS displays course material to the student. The program requires single key input by the student at the end of each screen. On completion of each lesson, the program records data to a disk file for analysis and lesson improvement.

WRITE/BAS generates the text and branching table files for use with LEARNER/BAS. A courseware author can create new lesson files, edit existing lesson files, print lesson files, and print student file reports.

Program documentation is complete so as to allow modifications and enhancements by the user (BASIC programming ability required) based on unique requirements or desires. The user's guide, although short, is complete and reflects the ease of program use.

END
DATE
FILMED
FEB.
1988